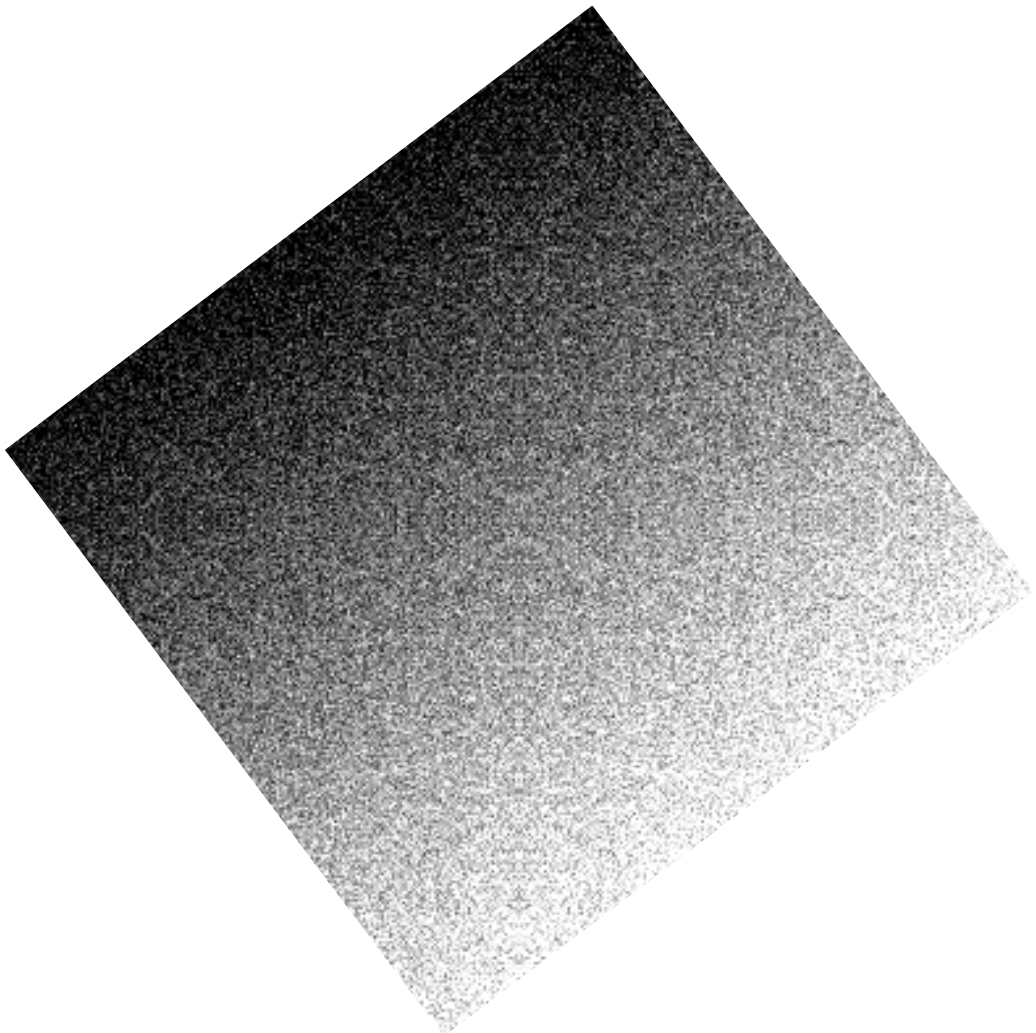
Intelligent Decision Server

IBM

Forms Design User's Guide

Version 1



SC26-8899-00

Intelligent Decision Server

IBM

Forms Design User's Guide

Version 1

SC26-8899-00

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page viii.

First Edition (October 1996)

This edition applies to Version 1 of Intelligent Decision Server, 5639–IDS, and to any subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1996. All rights reserved.**
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Notices . . . viii
Trademarks . . . viii

About This Book . . . x
Who Should Read This Book . . . x
Prerequisite Reading . . . x
We'd Like Your Comments . . . x

Chapter 1. What Is Forms Design? . . . 1
Discovering the Uses of Forms Design . . . 1
Gathering the Necessary Information . . . 1
Designing Forms . . . 2
Functioning as a Forms Designer . . . 3

Chapter 2. Getting Started . . . 5
Looking at the Forms Design and Data Entry Icons . . . 5
Configuring a Forms Design Icon . . . 5
Securing a Forms Design Icon . . . 8
Learning About the Forms Design Windows . . . 9
Understanding the Main Forms Design Window . . . 10
Depictions . . . 11
Window Header Buttons . . . 11
Special Menu Items . . . 12

Chapter 3. Performing the Basics . . . 13
Designing an Entry Form . . . 13
Constructing Retrieval and Update Forms . . . 15
Creating a Data Entry Icon . . . 15
Learning the Shortcuts . . . 16

Chapter 4. Designing the Overall Form . . . 18
Specifying Icon Options . . . 18
Creating Valid Combinations of Forms . . . 19
Setting Entry Form Options . . . 19
Specifying Retrieval Form Options . . . 21
Setting Update Form Options . . . 21

Chapter 5. Setting Preferences for Fields . . . 24
Accessing Field Options . . . 24
Understanding the Options . . . 26
Labeling Fields . . . 26
Designing Fields . . . 27
Choosing Validation Methods . . . 27
Initializing Fields . . . 27
Storing Information . . . 27

© Copyright IBM Corp. 1996

Chapter 6. Discovering Data Entry Methods 28
Defining Field Types 28
Displaying the Field Options Windows 30
Entering a Prompt Message 31
Creating Predefined Fields 31

Chapter 7. Constructing Entry Fields 33
Creating Entry Fields 33
Setting the Dimensions 35
Aligning Horizontally and Vertically 35
Associating with a Browse List 37
Figuring Out the Return Column 37
Displaying the Browse List 38
Entering the Browser Parameters 39

Chapter 8. Designing Choice Fields 40
Learning About Choice Fields 40
Creating Choice Fields 40
Adding Choice Field Choices 41
Naming Choices 42
Layering with Choice Fields 44
Creating Dependent Fields 45
Setting Preferences for Choice Fields 47
Presenting Choices 48
Setting the Choice Width 49
Setting the Choice Height 49
Deciding How to Define Choices 50
Defining the Choices Yourself 50
Generating the Choices Dynamically 50
Positioning Dependent Fields 52
Enhancing Retrieval Form Choice Fields 53

Chapter 9. Building Flexibility with the Section Field 54
Learning About Section Fields 54
Creating Section Fields 55
Adding Section Field Choices 55
Organizing Information into Sections 57
Setting Preferences for Section Fields 58
Presenting the Choices 58
Positioning Dependent Fields 59
Setting the Choice Width 60
Setting the Height of a Section Field 61

Chapter 10. Validating Data 62
Discovering Validation Options 62
Determining the Validation Method 64
Specifying Single or Multiple Entries in an Entry Field 64
Making Input Required or Optional 65

Specifying an Acceptable Data Type Value . . . 65
Creating Integer Value Ranges . . . 66
Creating Real Number Value Ranges . . . 66
Setting Character String Restrictions . . . 66
Creating Date Value Ranges . . . 68
Specifying Database Validation . . . 71
Setting No Database Validation . . . 71
Validating Data with a Browse List . . . 72
Preventing Duplicated Data . . . 72
Validating Data in a Database Table . . . 73
Entering an SQL Statement . . . 73
Using the Query Tool to Enter an SQL Statement . . . 73
Learning About the Default Validation . . . 74
Constraining Information Retrieval . . . 74

Chapter 11. Initializing Fields . . . 76
Finding Initialization Options . . . 76
Initializing Entry and Predefined Fields . . . 77
Selecting Initial Values for Entry Fields . . . 78
Setting Initial Values for Predefined Fields . . . 78
Discovering Initialization Methods . . . 78
Initializing Choice and Section Fields . . . 80
Learning Initialization Methods . . . 81
Changing Field Reset Options . . . 83
Learning About the Default Initialization . . . 84

Chapter 12. Storing Data . . . 85
Locating Storage Options . . . 85
Storing Data in the Database . . . 86
Specifying Entry or Predefined Field Storage . . . 87
Specifying Choice or Section Field Storage . . . 87
Specifying Database Storage for Choice or Section Fields . . . 87
Storing Data in an @-Variable . . . 88
Specifying No Data Storage . . . 89
Learning About the Default Storage . . . 89

Chapter 13. Working with the Browse Feature . . . 90
Setting Up Browse Lists . . . 90
Adding a Browse List to the Browser Library . . . 91
Setting Preferences for the Browse List . . . 92
Naming the Browse List . . . 93
Updating the Browse List . . . 93
Determining the Browser SQL . . . 94
Customizing Browse Lists . . . 96
Choosing a Method of Customizing . . . 96
Adding Columns to the Browse List . . . 96
Designing Browse Columns . . . 97
Predicting Column Length . . . 99

Ordering Columns 100
Determining Column Display 101
Entering a Formula 101
Setting Data Type 101
Using One Browse List for Several Fields 102
Converting Existing Forms Design Browse Lists 105
Returning Information from the Right Column 105
Understanding the Browser SQL Choice 105

Chapter 14. Enhancing Forms with Design Techniques 106
Grouping Fields 106
Creating Groups 106
Designing Groups 107
Placing Fields 109
Setting the Field Size 110
Setting Field Separation 111
Aligning Horizontally and Vertically 111
Using Lines to Separate Fields 111
Creating Lines 111
Changing the Line Type 112
Embedding Special Instructions 112

Chapter 15. Becoming an Expert Forms Designer 115
General Form Considerations 115
Using Primary Key Fields in Forms 115
Limiting the Number of Tables Each Form Represents 115
Using Views in Forms 116
Displaying Database Information for Fields 116
Working with Capsule Applications 117
Creating a Form to Set @-Variables 117
Verifying @-Variable Entries 119
Opening a Data Entry Icon Automatically 119
Creating a Forms Design Icon from a Data Entry Icon 120
Generating Post-Entry SQL 120
Inserting or Updating Data in More Than One Table 121
Accessing SQL Statements in a Source Document 122
Producing an Audit Trail 123
Constructing an SQL Constraint 123
Specifying Forms Design @-Keywords 125
Assigning a Global Name 125
Setting Up Conditional Expressions with @PARMs 126
Modifying Forms Design Tools with Scripting 126
Using Column and Table Name Indirection 128

Chapter 16. Designing Complex Forms 129
Structuring the Entry and Update Forms 131
Looking at the Employment History Section 132
Browsing Through the Departmental Section 133

Exploring the Personal/Benefits Section 134
Reviewing with the Performance Review Section 136
Creating the Retrieval Form 137

Chapter 17. Improving Security and Performance 139
Implementing Database Security 139
Assigning User IDs and Passwords 139
Predefining User Access 139
Assigning Access Keys 141
Predefining Database Table Access with the Secure Feature 141
Determining Data Entry SQL Performance 141
Generating Database-defined Choice Fields 141
Generating Database-validated Entry Fields 141
Generating Database-initialized Fields 142
Executing Commands in the SQL Control Window 142
Working with Production Tables 144

Appendix A. Identifying Keywords and Statements 145
Discovering the @-Keywords 145
Learning About &Statements 146
Knowing the Acceptable Values for Scripting Keywords 147

Glossary 150

IDS Publications 154

Index 155

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information that has been exchanged, should contact:

IBM Corporation
555 Bailey Avenue, W92/H3
P.O. Box 49023
San Jose, CA 95161–9023

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks

The following terms are trademarks of IBM Corporation in the United States or other countries or both:

ACF/VTAM	DXT	PROFS
AIX	Enterprise System/9000	PS/2
Application System/400	ES/9000	QMF
APPN	IBM	RACF
AS/400	MVS/ESA	RISC System/6000
Capsule	MVS/XA	RS/6000
DATABASE 2	OfficeVision	S/390

© Copyright IBM Corp. 1996

DataJoiner	OfficeVision/VM	SQL/DS
DB2	Operating System/2	System/360
DB2 for OS/2	Operating System/400	System/370
DB2/6000	OS/2	Virtual Machine/Enterprise Systems Architecture
DFSORT	OS/400	VM/ESA
Distributed Relational Database Architecture	Personal System/2	VTAM
DRDA	Presentation Manager	Win-OS/2

C-bus is a trademark of Corollary, Inc.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

About This Book

This book contains information about the components, features, and functions of the Forms Design tool.

Who Should Read This Book

This book is intended for users who are responsible for creating data entry forms. It assumes that you have a working knowledge of:

The IDS desktop environment

The Data Entry tool

SQL language

Prerequisite Reading

To understand the concepts in this book, read the IDS *Data Entry User's Guide*. This guide describes how to set values inside a capsule application automatically or to modify data in a database.

We'd Like Your Comments

Your feedback is important to us in providing you with the most accurate and high-quality information. If you have any comments about this book or any other IDS documentation, please visit our Website at:

http://www.dss.ibm.com

There you'll find the Feedback Connection page where you can enter comments and send them to us.

© Copyright IBM Corp. 1996

Chapter 1. What Is Forms Design?

Have you ever filled out a form, such as an application for employment or a credit card? If that application had been on line, it could have been a Data Entry form. Except, instead of your answers being read by the manager or credit agent, your answers are stored somewhere on line, perhaps in a database.

The Forms Design tool is used to create these on-line forms, or Data Entry forms. These forms are used to:

Enter data into the database

Retrieve, delete, and update data in the database

Set *@-variables* inside a capsule application

The Forms Design tool and Data Entry tool are separate from one another, but are closely related. Forms designers use the Forms Design tool to create the Data Entry tools that are used by others. This relationship helps to ensure security because forms designers can limit the user's access on a form-by-form basis.

Discovering the Uses of Forms Design

Within IDS, information can be stored in a database or in *@-variables* inside a capsule application. But wherever the information resides, you need a method of inserting it and retrieving it later. Data entry forms are the vehicles that allow you to accomplish this information transfer. As the forms designer, you create data entry forms with the Forms Design tool.

For example, suppose your Human Resources department wants to keep certain information on line for the employees at your company. Your system administrator has set up databases to store the information, and you need to create the data entry forms that allow Human Resources to interact with the databases.

Gathering the Necessary Information

As the forms designer, you must know the answers to several important questions before you begin to design a form:

What type of action needs to be performed?

You use different data entry forms for entering, retrieving, and updating information, so you must know in advance what action is to be performed. For example, the Human Resources department probably needs all three types of forms so that they can add employees, retrieve their information later, and make changes.

Does the form use a database or is the form for use with a capsule application?

You must know whether the form is meant to access a database or interact with a capsule application before designing the form. For example, the forms for the Human Resources department are meant to access a database.

© Copyright IBM Corp. 1996

What restrictions must be in place for the form?

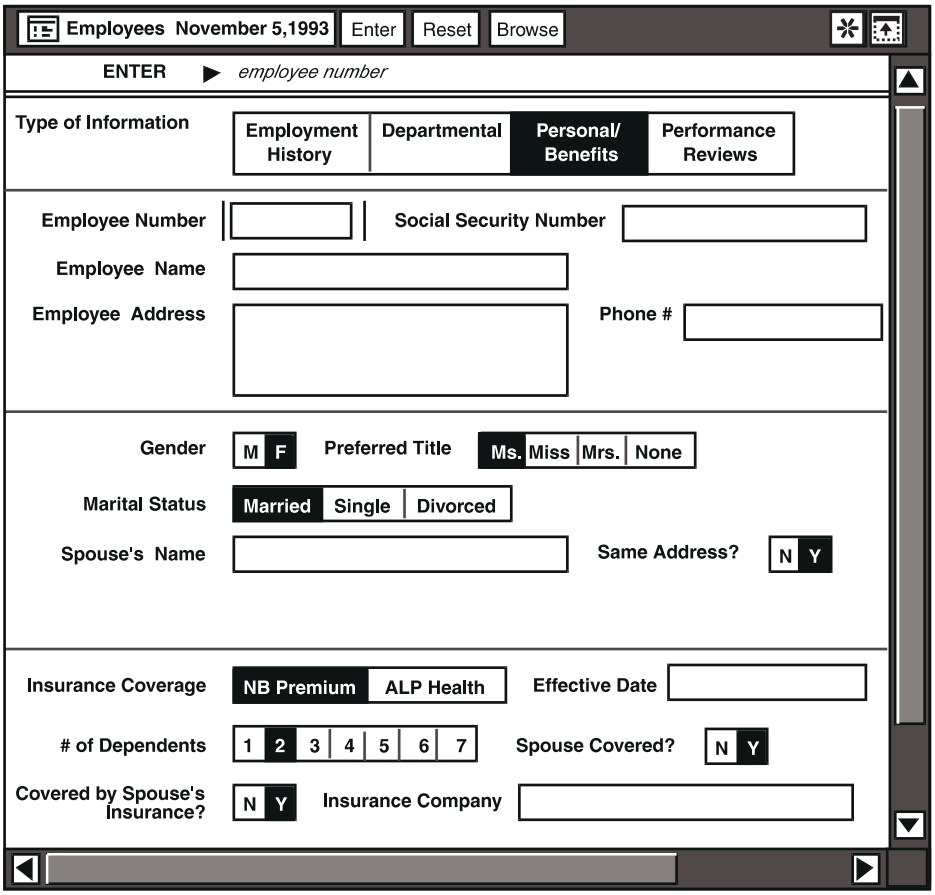
You need to know whether there is privileged information so that you can set up user access. For example, the forms for the Human Resources department should not be accessed by anyone outside of that department.

What information needs to be entered into the form?

You need to know what information is to be entered into the form so that you can design your form appropriately. For example, you need to interview the Human Resources department about what information they are planning to put into their databases, and come up with a structure for the data entry forms that makes logical sense to them.

Designing Forms

You design forms based on the answers to the previous questions. You can create fairly complex data entry forms by using the various features offered by the Forms Design tool. For example, when you have a form that needs to contain a great deal of information, you can use the features of the Forms Design tool to consolidate information and save space on your form. An example of a complex data entry form is shown in the following illustration.



Notice how the form consolidates space to show only the pertinent information. For example, in looking at the **Marital Status** field, you notice that because the **Married** choice is selected, the **Spouse's Name** and **Same Address?** fields appear. These fields are not necessary for an employee without a spouse, and therefore are not shown. The illustrated form saves space and structures information in many more ways than can be described here using the features of the Forms Design tool.

Functioning as a Forms Designer

As the forms designer, you need to understand the structure of the database, and be able to access all database information on which the proposed Data Entry icon operates. You determine which fields in the database can be accessed, and then design the forms to allow only that access.

Based upon your level of understanding, you might be asked to perform the following tasks:

Design fairly simple forms.

Design sophisticated forms, using your experience as a database administrator (DBA).

Design forms to set *@-variables* in capsule tools, using your experience in application development.

Chapter 2. Getting Started

Before you design a form, you should understand the basic features of the Forms Design tool as outlined in this chapter. The tasks associated with these basic features include:

Naming and configuring the Forms Design icon

Discovering the various windows of the Forms Design tool

Learning about the main Forms Design window

Understanding the types of forms you can create

By reading this chapter, you will become familiar with the basic components of the Forms Design tool. You should read this entire chapter before you read the remaining chapters of the guide.

Looking at the Forms Design and Data Entry Icons

The Forms Design and Data Entry icons look very similar. Both show a depiction of a form, but the Forms Design icon also shows a pencil creating a form. Data Entry icons must be created from Forms Design icons, so you will not find a Data Entry icon in the *Blank Icons file drawer.

To understand the Forms Design tool, you must first understand the Data Entry tool because it is the product of the Forms Design tool. Read and work through the IBM IDS *Data Entry User's Guide* before using the Forms Design tool.

Configuring a Forms Design Icon

After you copy a Forms Design icon from the *Blank Icons file drawer, you must configure it for a specific database before you can use it to access a database. To configure the icon, you must know the password and the name of the database or DB2 subsystem you want to access. If you do not have this information, consult your database administrator.

To configure a Forms Design icon:

1. Select the Forms Design icon and open the Icon Options window.
2. Select **Other Data** in the **Display** field.

   The **Other Data** settings are displayed in the window.

© Copyright IBM Corp. 1996

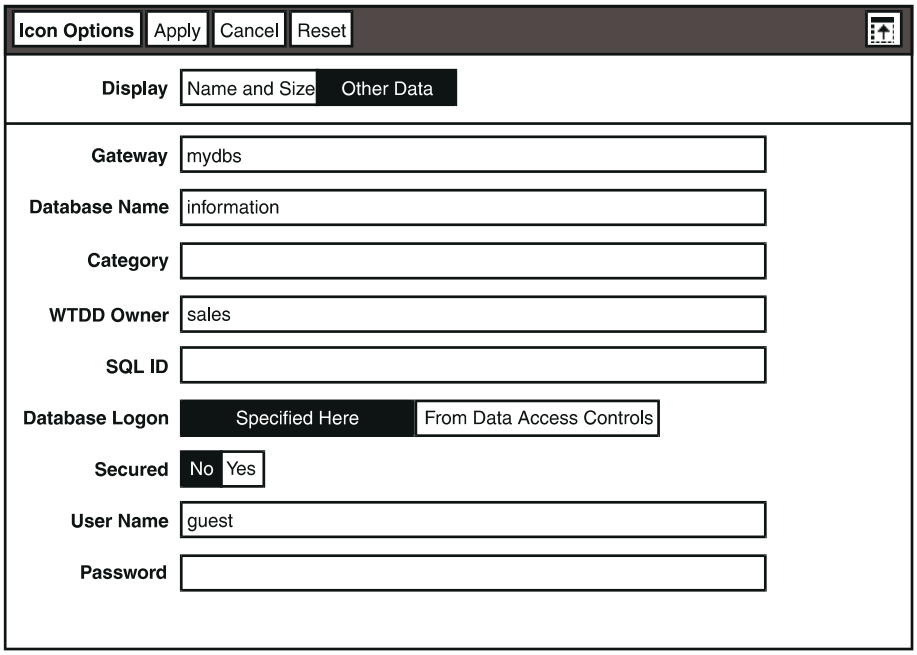


Figure 1. Icon Options window

3. Using the guidelines provided in Table 1, enter the required information into the text fields and select the appropriate choices in the other fields.

   When you create a Data Entry icon from a Forms Design icon, the Data Entry icon is configured automatically with the same information as the Forms Design icon. Therefore, it is good practice to configure your Forms Design icon to the name and password of the person who will be using it.

   When you create a Data Entry icon to be used inside a capsule application in such a way that it does not access the database, you do not have to configure the icon. See Chapter 15, “Becoming an Expert Forms Designer” on page 115, for more information on using Data Entry icons with capsule applications.

Table 1 (Page 1 of 3). Icon Options window settings

Field	Guidelines
Gateway	Type the name or address of the database gateway that you want the Forms Design and Data Entry icons to access. If the gateway service is in a different realm from the one you are using, immediately follow the gateway name with a colon and then the realm name. For example: mydbs:HQ

Table 1 (Page 2 of 3). Icon Options window settings

Field	Guidelines
Database Name	Type the name of the physical database you want the Forms Design and Data Entry icons to access. For DB2 databases, type the name of the DB2 subsystem you want to access. For SQL/DS databases, type the name of the virtual machine running SQL/DS. For AS/400 databases, it is not necessary to specify anything in this field, because the AS/400 is a single-database system. If you want to connect to a supported remote IBM database using Distributed Relational Database Architecture (DRDA), immediately follow the local database name with a period and then the name of the DRDA server through which the connection is being made. For example: DSN.TD1 If you want to connect to a remote Sybase** or Red Brick** database, immediately follow the SQL Server name with a vertical bar and then the name of the database you want to access. For example: SYBSERVER2|SYBPROD If you want to connect to a remote Oracle database, specify the full database name or its alias. For the full database name, specify the standard single-character prefix used for the remote connection (such as **T** for TCP/IP), followed by a colon, the name of the target host followed by another colon, and finally the system ID (SID) of the target Oracle database. For example: T:host1:DBS
Category	Type either the print name or internal name of the database category needed for this Forms Design icon. The name you specify here appears in the window header of the category window when the category is displayed.
WTDD Owner	If the Forms Design icon is configured to a database that supports qualified table names, type the name of the owner of the WTDD tables.
SQL ID	If you are using DB2 or DB2 for OSfile Database Manager, type the table qualifier name. The SQL ID overrides the default table qualifier name, which is the name entered in the **User Name** field. If you have questions about the table qualifier name, consult your database administrator.
Database Logon	Select **Specified Here** if you want database access to be based on information you type in the **User Name** and **Password** fields. Select **From Data Access Controls** if you want access to be based on information in the desktop's Data Access Controls window. Consult your database administrator if you have security questions.
Secured	Specify **Yes** if you want the icon's **User Name** field to be set to the user ID of the desktop on which it resides. See the next section, “Securing a Forms Design Icon” on page 8, for important information about securing an icon.

Table 1 (Page 3 of 3). Icon Options window settings

Field	Guidelines
User ID	Type a user ID the database can identify as valid. This might or might not be the same as your desktop user ID depending on how your database security is set up. When the Forms Design icon connects to the RDBMS, this is the user ID sent to the RDBMS. If you are using the OS/2 DRDA gateway to connect to a DRDA server, and your DB2 for OSfile user ID is different from your DRDA server user ID, you must specify both user IDs separated by a period. For example: DB2 for OSfile_userid.DRDAID
Password	Type the database user's password. This password might or might not be the same as your desktop password depending on how your database security is set up. For security, the password is never displayed. If you are using the OS2-DRDA gateway to connect to a DRDA server, and your DB2 for OSfile password is different from your DRDA server password, you must specify both passwords separated by a period. For example: DB2 for OSfile_password.DRDA_password

Securing a Forms Design Icon

When a secured icon is mailed, the User Name changes to the user ID of the recipient. The icon cannot be opened or used by the recipient unless his or her desktop name has the appropriate privileges for the database. You cannot change the User Name field of a secured icon.

If you receive a secured icon from another user and the database does not recognize your user ID, you will not be able to open the icon.

Important: Securing an icon is an irreversible operation. Before you secure an icon that you plan to distribute to other users, check with your database administrator to ensure that their desktop names are valid user IDs for the database.

An alternate way to secure a Forms Design icon is to configure it to read the user ID and password setting from the desktop's Data Access Controls window. For more information about the Data Access Controls window, refer to the IBM publication *Working with the Intelligent Decision Server*, or consult your system administrator.

Learning About the Forms Design Windows

The Forms Design tool operates in several windows that accomplish varied tasks. The logical order of the windows in relation to the main Forms Design window is shown in Figure 2.

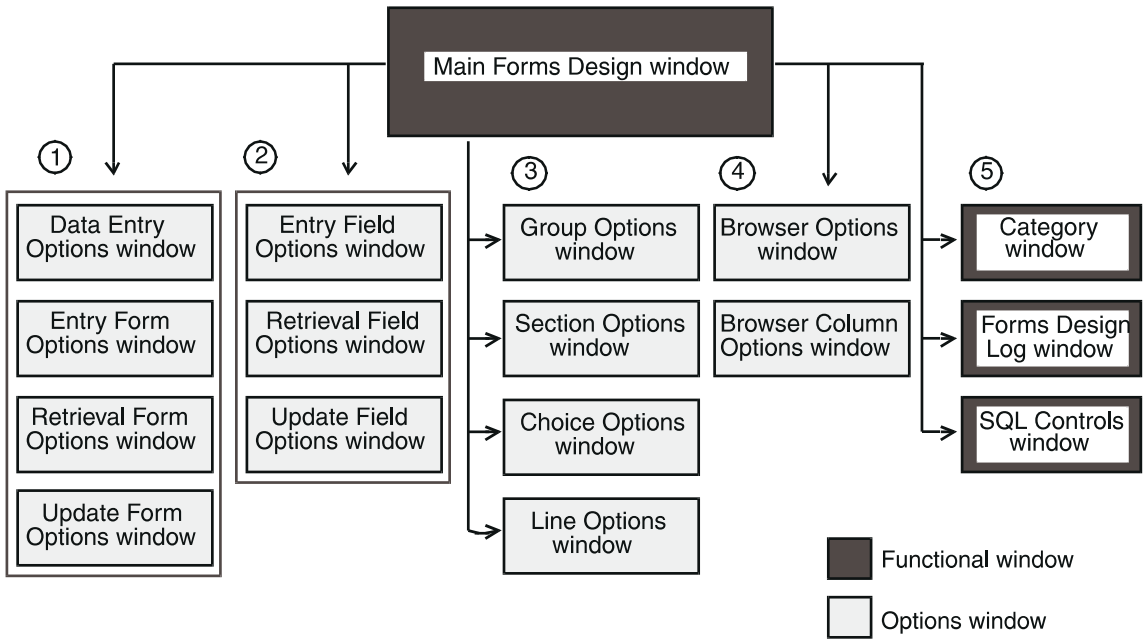


Figure 2. Diagram of the Forms Design windows

Figure 2 shows that there are five branches off the main window:

1. Options windows that apply to the entire data entry form. These options are described in Chapter 4, “Designing the Overall Form” on page 18.
2. Options windows that apply to specific fields within the form. These options are described in Chapter 5, “Setting Preferences for Fields” on page 24.
3. Options windows that apply to specific types of fields, such as a group of fields, a section field, a field, or a line field. These types of fields are explained in further detail in Chapter 6, “Discovering Data Entry Methods” on page 28, and Chapter 14, “Enhancing Forms with Design Techniques” on page 106.
4. Options windows that apply to browse lists. These options are described in Chapter 13, “Working with the Browse Feature” on page 90.
5. Functional windows that assist with the creation of a data entry form. The category window is described in Chapter 3, “Performing the Basics” on page 13. The other two windows are explained in Chapter 17, “Improving Security and Performance” on page 139.

Each window shown in Figure 2 is described in Table 2 on page 10.

Table 2. Forms Design windows

Name	Tasks accomplished in this window
Main Forms Design window	Construct forms and add browse lists.
Category window	View depictions of the tables in the specified database category. The database category is specified in the Icon Options window.
Data Entry Options window	Set options for the Data Entry icon that affect all of its forms.
Entry Form Options, Retrieval Form Options, and Update Form Options windows	Set options for the entry form, retrieval form, and update form that affect all fields in the specified form.
Entry Field Options, Retrieval Field Options, and Update Field Options windows	Set options for individual fields within a form.
Group Options window	Set options for a group of fields within a form.
Browser Options window	Set options for a browse list.
Browser Column Options window	Set options for a customized browse list column.
Section Options window	Set options for sections within a section field.
Choice Options window	Set options for choices within a field.
Line Options window	Set options for lines that divide a form into areas.
Forms Design Log window	View a record of all actions performed while the main Forms Design window is open.
SQL Controls window	Enter SQL statements that are sent to the database.

Understanding the Main Forms Design Window

When you open a Forms Design icon, the main Forms Design window appears on your desktop. Enlarge the window until it appears as shown in Figure 3.

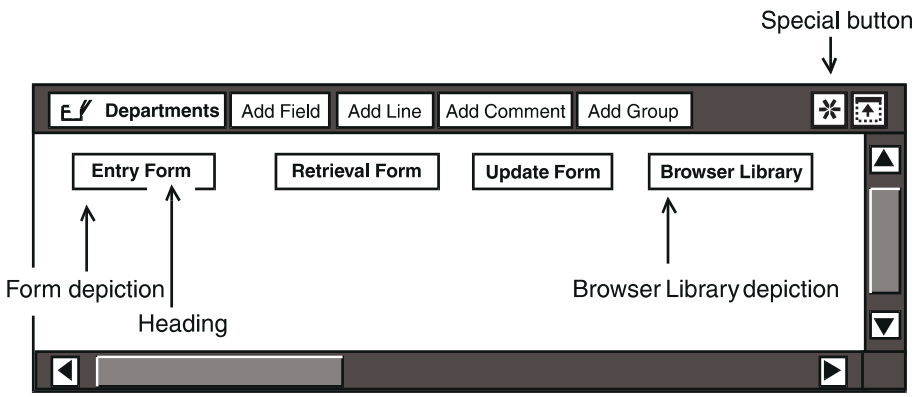


Figure 3. Main window of the Forms Design tool

The main Forms Design window contains three major elements:

Depictions

Window-header buttons

Special menu

These elements are discussed in the following sections.

Depictions

The main Forms Design window contains three form depictions, and a browser library depiction. The uses of the four depictions are explained below:

Entry Form

Defines the fields into which data can be entered. If you are creating a Data Entry icon to be used only for data entry, you only need the Entry Form depiction.

Retrieval Form

Defines the fields that allow retrieved rows to be viewed and deleted. If you are creating a Data Entry icon to be used only for data retrieval, you only need the Retrieval Form depiction.

Update Form

Defines the fields that can be updated. If you are creating a Data Entry icon to be used only for retrieving and updating data, and not for entering data, then you use both the Retrieval Form and the Update Form depictions.

Browser Library

Defines the browse lists that you specify.

Use all three form depictions if you are creating a form for entering, retrieving, and updating data.

Window Header Buttons

The Forms Design window header contains several buttons that are necessary for constructing forms. The buttons and their uses are described in Table 3.

Table 3 (Page 1 of 2). Main Forms Design window header buttons

Button Name	Function
Add Field	Adds a field to a form. Information on defining field options can be found in Chapter 6, “Discovering Data Entry Methods” on page 28.
Add Line	Adds a line to separate fields within a form. This feature is described in Chapter 14, “Enhancing Forms with Design Techniques” on page 106.
Add Comment	Adds special instructions or comments in a form. This feature is described in Chapter 14, “Enhancing Forms with Design Techniques” on page 106.

Table 3 (Page 2 of 2). Main Forms Design window header buttons

Button Name	Function
Add Group	Places fields side by side on a form by making them part of a group. This feature is described in Chapter 14, “Enhancing Forms with Design Techniques” on page 106.

Special Menu Items

When you click on the Special button in the Forms Design main window header, the Special menu opens.

The Special menu contains several items that are useful to the forms designer. Table 4 describes these items.

Table 4. Forms Design Special menu items

Item Name	Function
Show Data Entry Options	Displays the Data Entry Options window from which you can set options for the Data Entry icon that affect all the forms in the icon. See Chapter 4, “Designing the Overall Form” on page 18, for more information.
Create Data Entry Icon	Creates a Data Entry icon with the name of the Forms Design icon and the date it was created.
Show Category	Displays the category window which contains depictions of the tables in the database category you specified. See Chapter 3, “Performing the Basics” on page 13, for more information.
Capture SQL	Opens the SQL Controls window which logs the SQL activities. See Chapter 17, “Improving Security and Performance” on page 139, for more information.

Chapter 3. Performing the Basics

When you have configured a Forms Design icon, you can use it to create the forms that compose the Data Entry icon. This chapter describes the process for creating a simple Data Entry icon. By completing this example, you will learn how to:

Add fields to a form using the category window

Copy fields to other forms

Create a Data Entry icon

The example on the following pages gives you a basic understanding of the Forms Design tool. The remaining chapters of this guide expand on the information in this chapter by explaining the various options that are available to you when you design forms.

Designing an Entry Form

When designing forms, you can copy columns from a table in the category window into the depictions of the forms in the main window. The following example uses a sample database, but the instructions work for any database you decide to use. Keep in mind that the windows depicted will not match exactly those on your screen.

1. Open the Forms Design icon (named *Departments* in this example).
2. Select **Show Category** from the Special menu in the main window.

The category window opens as shown in Figure 4 on page 14.

© Copyright IBM Corp. 1996

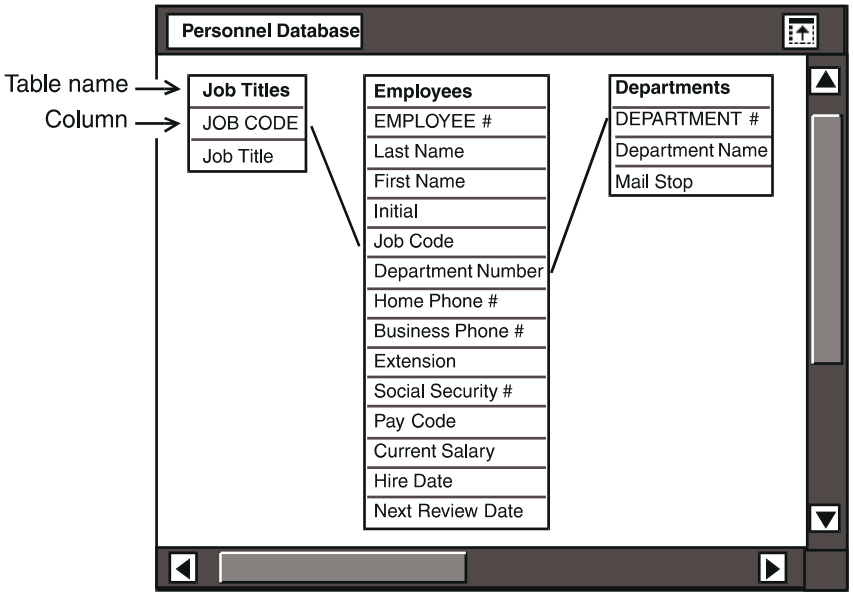


Figure 4. Category window

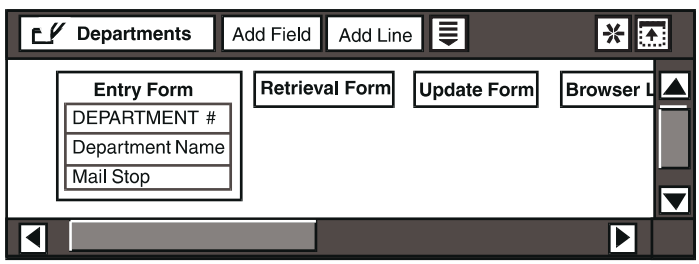
The category window shows the tables within the database category you specified in the Icon Options window when you configured the icon. In this example, the category window shown contains depictions of the tables in the Personnel Database category.

1. Select one or more columns in the category window.

   For this example, you could select the DEPARTMENT # column in the Departments table, and extend on the Department Name and Mail Stop columns.

2. Copy one or more columns to the heading of the Entry Form depiction in the main Forms Design window.

   The Entry Form depiction now contains the columns that you copied. These columns are displayed under the Entry Form heading in the main Forms Design window. Notice that the form depiction expands to accommodate the length of the column names.



3. Close the category window.

Constructing Retrieval and Update Forms

After you create fields in one form you can then copy and move the fields within the form, or between form depictions in the main window.

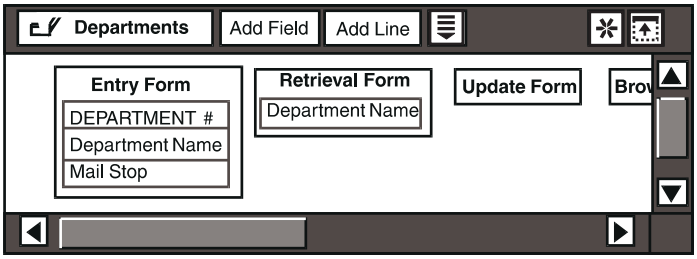
To create retrieval and update forms:

1. Select a field in the Entry Form depiction.

   In this example, select the **Department Name** field.

2. Copy it to the heading of the Retrieval Form depiction.

   The field you copied now appears in the Retrieval Form depiction.

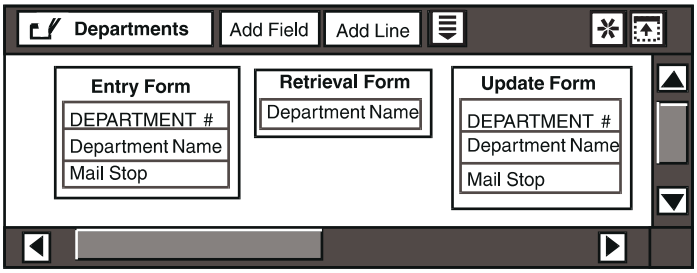


3. Select another field in the Entry Form depiction.

   For example, select the **DEPARTMENT #** field, and extend on the **Department Name** and **Mail Stop** fields.

4. Copy these fields to the heading of the Update Form depiction.

   The fields that you copied now appear in the Update Form depiction.



You are now ready to create a Data Entry icon with the forms that you have created in the main Forms Design window.

Creating a Data Entry Icon

To create a Data Entry icon:

1. Select **Create Data Entry Icon** in the Special menu in the main Forms Design window.

   When you select **Create Data Entry Icon**, the pointer changes, and the following message appears in the message area of your desktop:

   Select location of new Data Entry icon.

2. Select an empty location on your desktop or in a container, such as a folder.

   A Data Entry icon now appears in that position on your desktop. The name of the Data Entry icon assumes the name of the Forms Design icon that created it, along with the date that it was created. For example, if you create a Data Entry icon from the Forms Design icon used in the previous examples, the name of the Data Entry icon is *Departments*, followed by the date the icon was created.

3. Open the Data Entry icon.

   In this case, the form can now be used to enter, retrieve, and update information in the Departments table.

Learning the Shortcuts

There are several methods of creating data entry forms that can save you time and effort. The following are a few of these methods:

Know where your data is being stored before you create fields.

Deciding whether to copy fields from columns in the category window or define them yourself depends on where you plan to store the data from the Data Entry icon. If you are creating a Data Entry icon to store data in the database, copy your fields from columns in the category window. By using this method, you eliminate the need to specify database information because fields copied from the category window already have this information.

However, if you are creating a Data Entry icon to store data in the User Input Control window of a capsule application, then create fields by clicking on the **Add Field** button in the main Forms Design window header. See Chapter 12, “Storing Data” on page 85, for more information on storing data in *@-variables*.

Design the field options before you copy fields between forms.

In this chapter, you selected fields from the Entry Form depiction and copied them to the Retrieval Form and Update Form depictions without defining any field options. You can save time by first setting the field options for one form, then copying the fields into the other forms that you are creating. By using this method, you do not have to redefine each field as you do if you copy columns repeatedly from the category window. See Chapter 5, “Setting Preferences for Fields” on page 24, for more information about setting field options.

Utilize Forms Design's more technical features to reduce form complexity.

In most cases, you create forms that contain fields from only one database table. However, you can have a form depiction that contains fields from more than one table, or you can specify one or more SQL statements that run after data is

inserted or updated in a table. See Chapter 15, “Becoming an Expert Forms Designer” on page 115, for more information.

Chapter 4. Designing the Overall Form

The first step in designing a Data Entry form is to set options for the Data Entry icon that apply to all the forms in the icon. You can also set options for the individual forms within the icon, such as the entry form, retrieval form, and update form. This chapter describes the options for the:

Data Entry icon

Entry form

Retrieval form

Update form

The options described in this chapter are *global*, meaning they affect the entire form or icon. The Forms Design tool also enables you to set *local* options; for example, options for specific fields within a form. See Chapter 5, “Setting Preferences for Fields” on page 24, for information on setting options for specific fields.

Specifying Icon Options

With the Data Entry Options window, you can set several options for the Data Entry icon that affect all of the forms in the icon. The entry form, retrieval form, and update form are all affected by the options set here.

To display the Data Entry icon options:

1. Select **Show Data Entry Options** in the Forms Design Special menu.

The Data Entry Options window opens.

You can set the following options within the Data Entry Options window:

Error signal

Determine the action taken by the system if incorrect information is entered into a field by making a choice in the **Data Entry Error Signal** field. You can specify that either an error message flashes or a beep sounds. If you specify that the error message is an audio signal, you can control its duration.

If you specify that the error message is an audio signal, a sample beep of the specified duration sounds when you close the Data Entry Options window, or when you click on the **Apply** button.

Name of the Data Entry icon

Specify the name that is given to the Data Entry icon by the Forms Design icon within the **Data Entry Icon Name** field. The default name is the name of the Forms Design icon and the date. See Chapter 15, “Becoming an Expert Forms Designer” on page 115, for information about the keywords that you can use to generate the icon's name.

© Copyright IBM Corp. 1996

Precision of real numbers

Set the default precision used for all real numbers in the Data Entry forms within the **Real Number Precision** field. The default is 10.2, which indicates that the forms can accommodate entries that include 10 numerals with a decimal precision of 2. For example: 98765432.10.

Be sure that the precision is set correctly for the data that is entered into the forms. If the precision is not set correctly, the user receives ##### instead of data in a retrieval form.

The Forms Design tool adheres to precision settings made in the desktop Set Preferences window by default. However, if you set the precision of real numbers with the Data Entry Options window, these settings override those made in Set Preferences window.

Creating Valid Combinations of Forms

Although you can create various combinations of forms for each Data Entry icon, not all combinations of forms are actually valid. For example, if a Data Entry icon contains only an update form, the user cannot update data because a retrieval form is needed to access the update form.

Table 5 shows:

The valid combinations of forms possible in a Data Entry icon

The types of functions allowed by different forms

Table 5. Valid combinations of forms

Data entry functions	Entry form	Retrieval form	Update form
Enter data	X		
Enter, retrieve, and delete data	X	X	
Enter, retrieve, delete, and update data	X	X	X
Retrieve, delete, and update data		X	X
Retrieve and delete data		X	

Setting Entry Form Options

You can set several options that affect both the appearance and operation of the entire entry form.

To display the entry form options:

1. Select the heading of the **Entry Form** depiction in the main Forms Design window.

   The depiction becomes highlighted.

2. Open the Entry Form Options window. The default options are displayed as shown in Figure 5 on page 20.

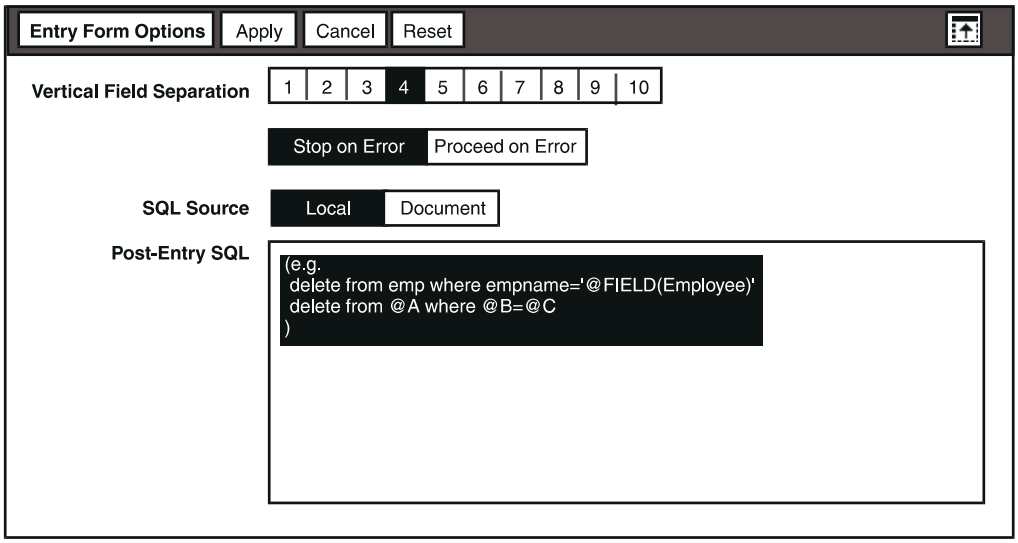


Figure 5. Entry Form Options window

You can set the following options for the entry form:

Vertical separation between fields

Set the relative amount of vertical space between fields in the Data Entry window by making a choice within the **Vertical Field Separation** field.

Behavior on error detection

Specify whether the Data Entry icon should stop if an error occurs when an SQL statement is run, or continue sending any remaining SQL statements.

The &ERRNUM statement in post-entry SQL supersedes the **Stop on Error**/**Proceed on Error** selection. For information about the statements and functions in post-entry SQL, see Appendix A, “Identifying Keywords and Statements” on page 145.

Source for SQL statements

Determine the location of post-entry SQL in the **SQL Source** field. You can use an existing SQL program in another document as the source for post-entry SQL, or enter specific SQL statements into the **Post-Entry SQL** field.

When you select **Local** as the source, the Entry Form Options window appears as shown in Figure 5. Provide SQL statements to be run each time the **Enter** button is clicked on in the Data Entry window. Enter these statements into the **Post-Entry SQL** field.

When you select **Document** as the source, a new field appears.

Enter the name or path reference to an SQL source document in the **Document Name** field.

See Chapter 15, “Becoming an Expert Forms Designer” on page 115, for more information about specifying the **SQL Source** and post-entry SQL options.

Specifying Retrieval Form Options

You can set several options that affect both the appearance and operation of the entire retrieval form.

To display the retrieval form options:

1. Select the heading of the Retrieval Form depiction in the main Forms Design window.

   The depiction becomes highlighted.

2. Open the Retrieval Form Options window. The default options are displayed as shown in Figure 6.

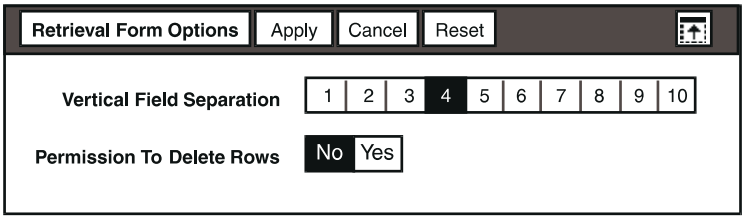


Figure 6. Retrieval Form Options window

You can set the following options for the retrieval form:

Vertical separation between fields

Set the relative amount of vertical space between fields in the retrieval form by making a choice within the **Vertical Field Separation** field.

Permission to delete rows

Specify whether or not the user can delete rows from the database by making a choice within the **Permission To Delete Rows** field. The user name and password entered in the Icon Options window must have appropriate privileges to delete rows from a database.

Use this option with caution because there are limited situations where you want to allow a user this privilege.

Setting Update Form Options

You can set several options that affect both the appearance and operation of the entire update form.

To display the update form options:

1. Select the heading of the Update Form depiction in the main Forms Design window.

   The depiction becomes highlighted.

2. Open the Update Form Options window. The default options are displayed as shown in Figure 7.

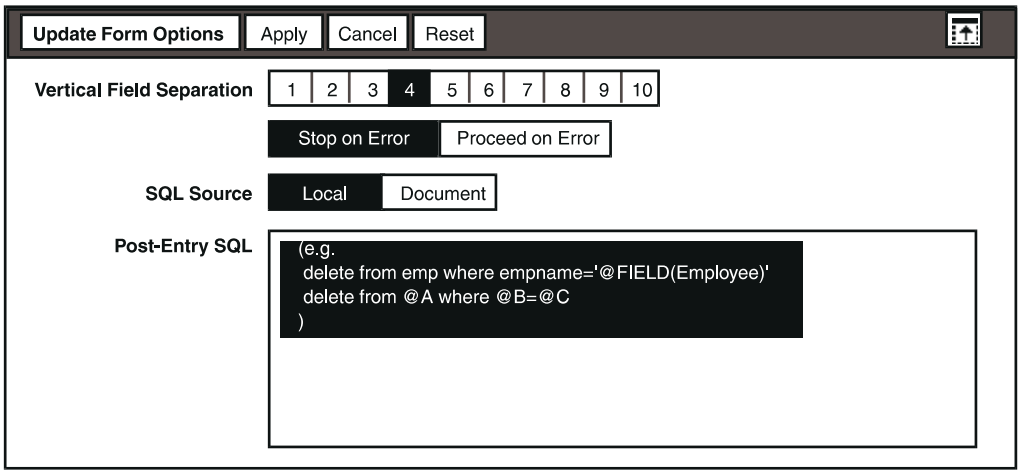


Figure 7. Update Form Options window

You can set the following options for the update form:

Vertical separation between fields

Set the relative amount of vertical space between fields in the update form by making a choice within the **Vertical Field Separation** field.

Behavior on error detection

Specify whether the Data Entry icon should stop if an error occurs when an SQL statement is run, or continue sending any remaining SQL statements.

The &ERRNUM statement in post-entry SQL supersedes the **Stop on Error**/**Proceed on Error** selection. For information about the statements and functions in post-entry SQL, see Appendix A, “Identifying Keywords and Statements” on page 145.

Source for SQL statements

Determine the location of post-entry SQL in the **SQL Source** field. You can use an existing SQL program in another document as the source for post-entry SQL, or enter SQL statements into the **Post-Entry SQL** field.

When you select **Local** as the source, the Update Form Options window appears as shown in Figure 7. Provide SQL statements to be run each time the **Enter**

button is clicked on in the Data Entry window. Enter these statements into the **Post-Entry SQL** field.

When you select **Document** as the source, a new field appears as shown in the following illustration.

Enter the name or path reference to an SQL source document in the **Document Name** field.

Chapter 5. Setting Preferences for Fields

The Forms Design tool enables you to design fields by setting *local options*, or options for specific fields that determine the appearance and functionality of those fields. This chapter provides:

A brief overview of the types of options that you can set for fields

Directions for accessing those options through the field options window

Although this chapter focuses on the Update Field Options window. The following chapters of this guide contain more detailed information about using the options described in this chapter.

Accessing Field Options

Each field in a form has unique options associated with it. You can set options to determine the following:

How the field appears in the Data Entry tool

How data is entered into the field

Where the data is stored

When you add a field to a form, the Forms Design tool assigns default options to that field. You can change most of these defaults through the field options window for the specific field.

The field options window for a field in the entry form is the Entry Field Options window, and the field options window for a field in the retrieval form is the Retrieval Field Options window.

The field options window differs depending upon the form in which the field resides, the type of data copied from the category window, and the choices selected within the fields of the field options window.

To change field defaults:

1. Select the field for which you want to set options. The field becomes highlighted.
2. Open the field options window. The field options window for a field in the entry form appears as shown in Figure 8 on page 25.

   Because the form you are creating is unique, the field options window on your desktop might appear slightly different than the Entry Field Options window illustrated in Figure 8 on page 25. Further, Figure 8 on page 25 shows the entire window, but you only see a partial window on your desktop due to the length of the form.

   When you select **Document** as the source, a new field appears.

© Copyright IBM Corp. 1996

3. Enter the name or path reference to an SQL source document in the **Document Name** field.

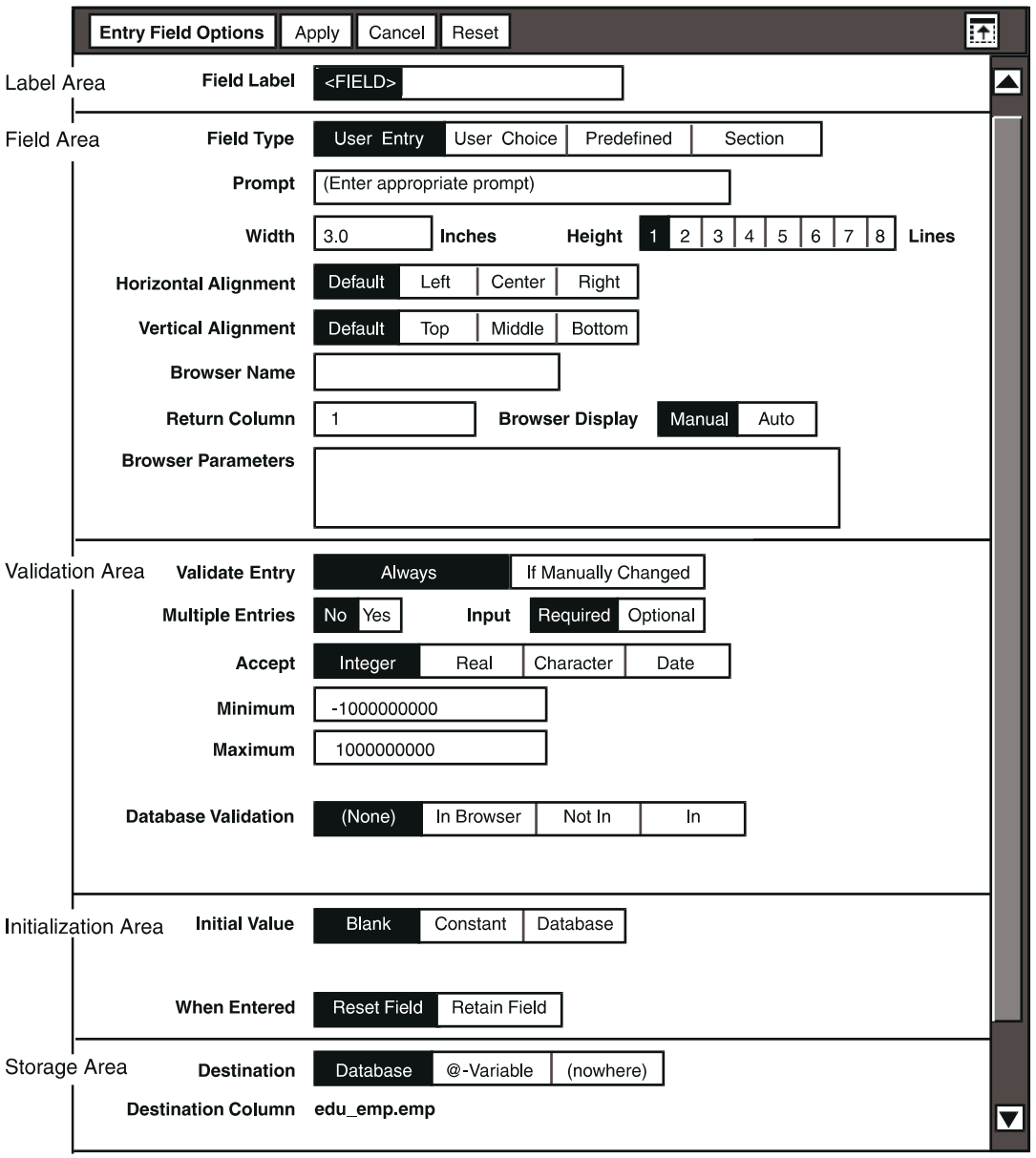


Figure 8. Entry Field Options window

4. Change the options as needed. The following section describes the options available to you.

5. Close the field options window.

Understanding the Options

Several lines divide the field options window into five areas, as was shown in Figure 8 on page 25:

Label

Field

Validation

Initialization

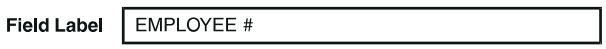
Storage

The first two areas, label and field, always appear in the field options windows. One or more of the three remaining areas appear depending upon the type of field that you select in the **Field Type** field, and the form in which your field resides. However, the general format of the field options windows is the same regardless of which choice in the **Field Type** field you select.

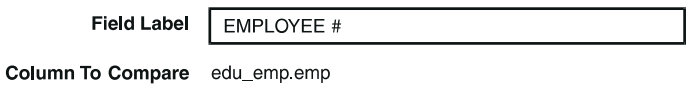
Labeling Fields

When you copy fields from the category window, the Forms Design tool initializes the field label to the print name of the database column to which it corresponds. Change the contents of the **Field Label** field if you want to change the name of the field in the Data Entry window.

The following illustration shows the label area for a field within a Data Entry form that contains employee information.



For a field within a retrieval form, the field area has an additional field, **Column To Compare**, that contains the name of the database column that corresponds to your field. The Data Entry tool compares the entry to the database column shown in the **Column To Compare** field.



The **Field Label** field is also where you enter a global name. To do so, type GLOBAL: followed by the global name that you are assigning to the field. See Chapter 15, “Becoming an Expert Forms Designer” on page 115, for information on @-keywords and the global function.

Designing Fields

Decide in the field area how data is entered into the field, and how the field appears in the Data Entry window.

For a description of the different types of fields available to you, and the options for a predefined field, see Chapter 6, “Discovering Data Entry Methods” on page 28. For information about field options for the other field types, see Chapter 7, “Constructing Entry Fields” on page 33, Chapter 8, “Designing Choice Fields” on page 40, and Chapter 9, “Building Flexibility with the Section Field” on page 54.

Choosing Validation Methods

The validation area appears only for fields. This area contains information about how the Data Entry tool validates data entered into a field. It also contains editing information if the field accepts character strings.

See Chapter 10, “Validating Data” on page 62, for a detailed description of validating field data.

Initializing Fields

Specify how the Data Entry tool *initializes*, or gives value to a field when the Data Entry window opens, by setting options in the initialization area. The initial value can be acquired from the database or an @variable in the User Input Control window of a capsule application, or you can specify a constant in the initialization area.

See Chapter 11, “Initializing Fields” on page 76, for a detailed description of the various initialization options.

Storing Information

Determine where the Data Entry tool stores the data from this field within the storage area. See Chapter 12, “Storing Data” on page 85, for a detailed description of this area.

Chapter 6. Discovering Data Entry Methods

You can specify several ways that data can be entered into a field in a Data Entry form. This chapter describes the ways that you can define a field, and explains the differences between the methods. The options discussed in this chapter appear in the field area of all three field options windows. The following chapters explain the options for the various data entry methods in greater detail.

See Chapter 5, “Setting Preferences for Fields” on page 24, for a complete overview of the sections in the field options windows.

Defining Field Types

Data can be entered into a field using four different field types. Figure 9 shows how each field type appears in an entry form.

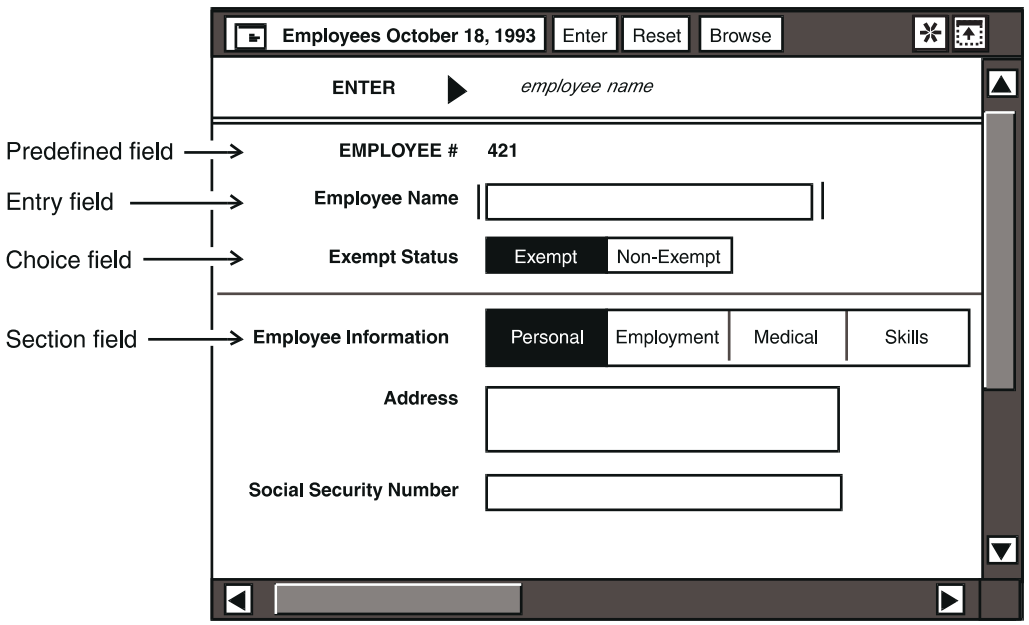


Figure 9. Predefined, entry, choice, and section fields

The Data Entry tool supplies a value that cannot be changed.

A field could be defined as any of the following:

Predefined field

Entry field

The user enters the data directly into the field, or selects a value from a browse list. The **User Entry** choice is the default of the **Field Type** field.

© Copyright IBM Corp. 1996

Choice field

The user selects a choice from the field. You can either define the choices yourself, or generate the choices dynamically from the database. Choice fields can also have *dependent fields*, or fields that appear only when a certain choice is selected. Choices made from dependent fields of a field must be applied while still viewing that dependent field.

Section field

The user selects a choice from the field that you have created. Each choice displays a section that contains dependent fields. When the user enters the information requested in each section and clicks on the **Enter** button, all sections are validated, including all dependent fields. A symbol, §, distinguishes a section field from a field.

Table 6 describes the criteria used to determine the field types for the Data Entry icon shown in Figure 9 on page 28. Use this table in conjunction with Figure 9 on page 28 to help you understand how to choose an appropriate field type.

Table 6 (Page 1 of 2). Explanations of field type decisions for Figure 5-1

Example Field	Field Type	Reason this field type was used:
EMPLOYEE #	Predefined	The user should not be able to change the employee number.
		The employee number can be determined from the name entered in the **Employee Name** field.
Employee Name	Entry	The user either knows the name to enter, or can find it in a browse list.
		Because the data entry form pertains to more than one employee, the field cannot be predefined.
		There are too many employees to create a field with all of their names.
		The choice of names alters due to changes in employment, so a field is not feasible.
Exempt Status	Choice	There are few choices for the field.
		The choices will not change.
		The descriptive names associated with each choice are short enough to display in a field.
		Only one choice needs to be applied at once.

Table 6 (Page 2 of 2). Explanations of field type decisions for Figure 5-1

Example Field	Field Type	Reason this field type was used:
Employee Information	Section	The information to be obtained can be organized into distinct sections which all have dependent fields. There is too much information for a field to be useful. The categories within the section will not change. All choices and their dependent fields are active and can be validated when the **Enter** button is clicked on. Therefore, all of the information for an employee can be entered at once without having to apply choices for each dependent field.
Address	Entry	The address cannot be predefined for new employees.
Social Security Number	Entry	The social security number cannot be predefined for new employees.

Displaying the Field Options Windows

The fields shown in the entry form in Figure 9 on page 28 were created in the **Field Type** field in the Entry Field Options window. The following steps explain how to access the Entry Field Options window. To open the Retrieval Field Options window or the Update Field Options window, simply select a field under the appropriate form depiction and follow these directions.

To display the Entry Field Options window:

1. Select a field in the **Entry Form** depiction of the main Forms Design window.
2. Open the Entry Field Options window.

   The fields that appear in an Entry Field Options window are determined by the type of data copied from the category window and the choices selected within the fields of the Entry Field Options window. Because the form you are creating is unique, the Entry Field Options window on your desktop might appear slightly different than the windows illustrated in this document.

   When you first copy a field from the category window into a form in the main Forms Design window, the Forms Design tool defines that field as an entry field. As the forms designer, you can change the field type after you have copied the field into a form.
3. Set the options to define your field.
4. Close the Entry Field Options window to apply your settings.

   The options in the field area change depending upon the selection you make in the **Field Type** field. The **Field Type** field and the **Prompt** field remain consistent for all field types but the predefined field type. The **Prompt** field is discussed in the following section. Except for the fields relating to the predefined field type, which are discussed in this chapter, all other fields relating to the field type chosen are explained in subsequent chapters.

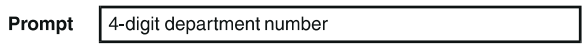
Entering a Prompt Message

When you choose to create an entry, choice or section field, you have the option of entering a prompt into the **Prompt** field. Enter information in the **Prompt** field that gives instructions applicable to the form being used.

When you first open the field options window for a field that was copied from the category window, a message already exists in the **Prompt** field. The WTDD (Workstation Tools Data Dictionary) provides an initial prompt based on the description field of the database column copied from the category window.

For example, suppose you copy a **DEPARTMENT #** field from the category window into the main Forms Design window. When you display the field options window for the **DEPARTMENT #** field, the prompt message could be dept number, as shown in the following illustration.

You can change the prompt message to be more specific, and thus more easily understood. For example, change the dept number prompt message to 4-digit department number.



When the **DEPARTMENT #** field is selected, the new prompt message appears as follows:

ENTER 4-digit department number

You can open the field options window and change the prompt message at any time. After each change you must create a new Data Entry icon with the updated prompt.

Creating Predefined Fields

When you define a field as predefined, the Data Entry icon supplies a value for the field. The value in a predefined field cannot be changed by the user. Examples of predefined fields are those that contain the current date or the name of the user.

To create a predefined field:

1. Copy a field from the category window to the form depiction in the main Forms Design window.
2. Open the field options window.
3. Select **Predefined** in the **Field Type** field.
4. Close the field options window.

When you define a field as predefined, you must specify the predefined value in the initialization area of the field options window.

Using predefined fields with **Blank** selected in the **Initial Value** field is useful for adding blank lines to a form. Blank space can enhance the appearance of a form because it makes fields stand out more clearly.

Using predefined fields with **Constant** selected in the **Initial Value** field is useful for adding comments or instructions to a form.

For more information about the initialization area of the field options window, see Chapter 11, “Initializing Fields” on page 76.

Chapter 7. Constructing Entry Fields

You can set many options for a field that determine its appearance and execution. The options discussed in this chapter appear in the field area of the field options windows when you select the **User Entry** choice in the **Field Type** field. This chapter describes:

- Setting width and height measurements
- Aligning the field horizontally and vertically
- Associating a browse list
- Specifying the return column for the browse list
- Displaying the browse list
- Setting parameters for the browse list

See Chapter 6, “Discovering Data Entry Methods” on page 28, for an overview of the different field types available to you. For information about options you can set in the other sections of the field options windows, see Chapter 10, “Validating Data” on page 62, Chapter 11, “Initializing Fields” on page 76, and Chapter 12, “Storing Data” on page 85.

Creating Entry Fields

You determine the type of field that you want to create within a form in the field options windows. Define a field as a field when you want the user to either enter a value into a field, or select a value from a browse list that contains valid values.

Examples of fields are name and address fields where the user enters a value or selects a value from a browse list. For example, you might want to create an entry form similar to that shown in Figure 10. In this example, the **Employee #** and **Employee Address** fields are fields.

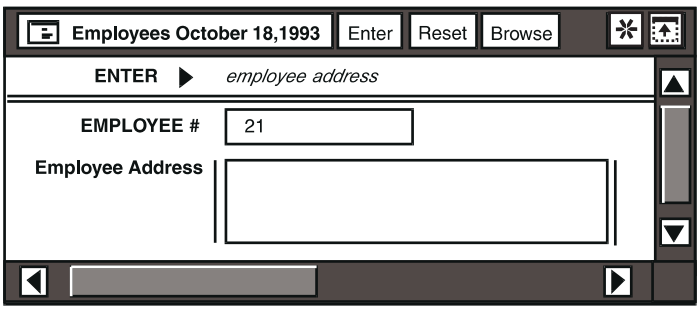


Figure 10. Entry fields in an entry form

You can create fields for either of the three form types (entry, retrieval, and update) by following the steps described below. Just substitute the correct field options window in the instructions.

© Copyright IBM Corp. 1996

To create a field for an entry form:

1. Select a field in the Entry Form depiction in the main Forms Design window.
2. Open the Entry Field Options window.
3. Select **User Entry** in the **Field Type** field.
4. Select a choice for each field and click on the **Apply** button.

   If you want a field to have a browse list, or if you want the field data to be validated against the database, you must also set certain options in the validation area of the field options windows. See Chapter 10, “Validating Data” on page 62, for more information.
5. Close the Entry Field Options window when complete.

All of the options in the field area of the Entry Field Options window apply to entry, retrieval, and update forms.

Table 7 (Page 1 of 2). Entry field options

Field	Choices	Function
Field Type	User Entry	Displays the options for a field.
Prompt	N/A	Prompts the user to perform an action, such as enter a value. The **Prompt** field can be used to clarify the format of the entry. The Forms Design tool initializes this field to the WTDD description of the column copied from the category window. For more information on redefining the prompt, see Chapter 6, “Discovering Data Entry Methods” on page 28.
Width	N/A	Sets the field width, in inches or centimeters, where data is entered.
Height (Lines)	1 to 8	Determines the number of lines of text allowed in the field.
Horizontal Alignment	Default	Aligns the field according to the system default.
	Left	Aligns the field flush left in its space.
	Center	Centers the field horizontally in its space.
	Right	Aligns the field flush right in its space.
Vertical Alignment	Default	Aligns the field according to the system default.
	Top	Aligns the field at the top of its space.
	Middle	Centers the field vertically in its space.
	Bottom	Aligns the field at the bottom of its space.
Browser Name	N/A	Specifies the browse list to be associated with the field.
Return Column	N/A	Specifies the column in the browse list whose values are to be returned to the field.
Browser Display	Manual	Displays the browse list when the **Browse** button is clicked on.
	Auto	Displays the browse list automatically when the user selects in the field.

Table 7 (Page 2 of 2). Entry field options

Field	Choices	Function
Browser Parameters	N/A	Specifies terms that can be used as constraints for the browse list.

Setting the Dimensions

Determine the dimensions of a field by entering information in the **Width** field and selecting a choice in the **Height (Lines)** field.

The characteristics of the **Width** and **Height (Lines)** fields are as follows:

The width of a field is measured in inches or centimeters, based upon selections in the Set Preferences window. Refer to the IDS publication *Working with the Intelligent Decision Server* for more information.

The height of a field is measured relatively by lines of text.

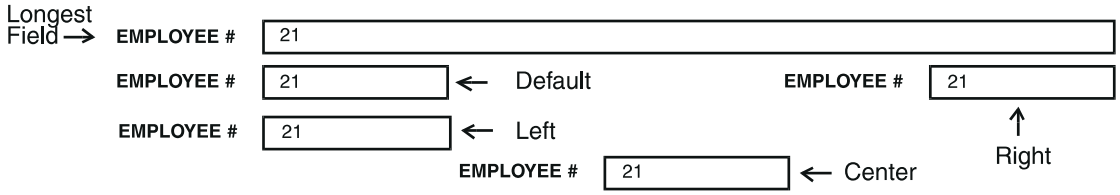
To determine the dimension of a field, enter the appropriate number of inches or centimeters in the **Width** field, and select the number of lines from the **Height (Lines)** field.

Aligning Horizontally and Vertically

You can determine the horizontal and vertical alignment of a field in a form by selecting choices in the **Horizontal Alignment** field and **Vertical Alignment** field.

To set the horizontal alignment of your field, place it either to the left, right, or center of its space, or choose the default setting. The horizontal space a field can occupy is determined by the longest field in the group or form. A *group* is a collection of fields that are bound together. For example, if you want to align several fields horizontally, you would put them in a group. See the section “Grouping Fields” on page 106 for more information about creating groups.

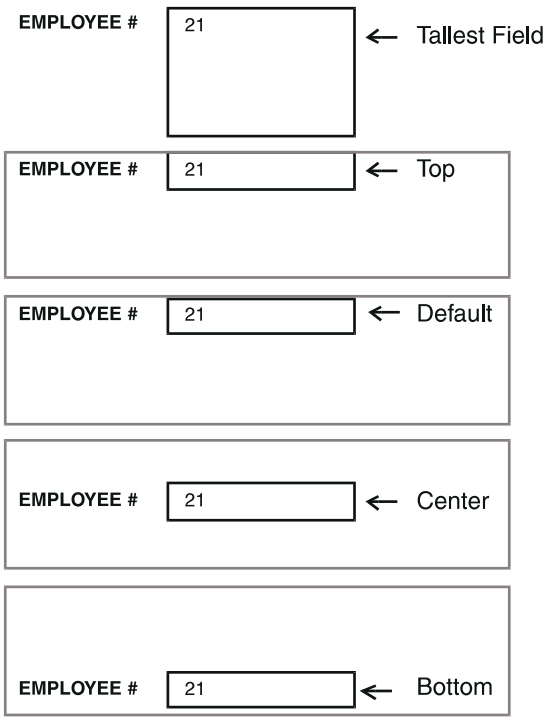
The following diagram illustrates the horizontal alignments available to you for a field relative to the space in which it resides.



You must place fields in a group to vertically align them. To set the vertical alignment of your field, place it at the top, middle, or bottom of its space, or choose the default setting. The vertical space a field can occupy is determined by the field with the greatest height in the group. To make this feature work within a group, you must

specify that the field size is constant. See the section “Setting the Field Size” on page 110 for more information about setting the size of fields within a group.

The following diagram illustrates the vertical alignments available to you for a field relative to the space in which it resides. The dotted lines represent the space each field can occupy.



When you specify the alignment of a field, however, you determine the horizontal and the vertical alignment together. For example, if you set the horizontal alignment to left, and the vertical alignment to top, the field aligns as shown in Figure 11. The dotted lines represent the space determined by the longest field and the tallest field. Note that the dotted lines do not appear on the screen.

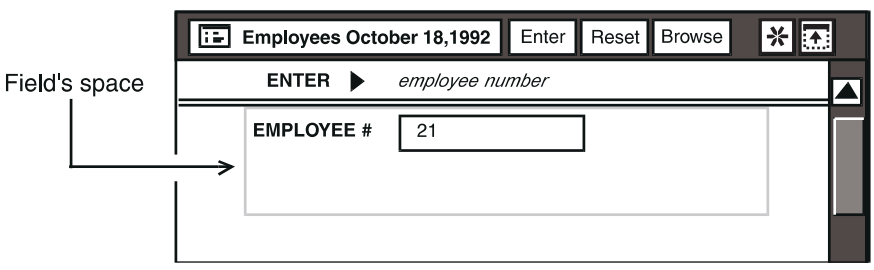
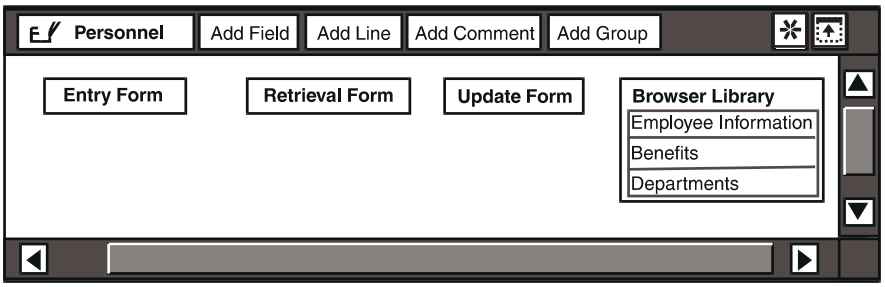


Figure 11. Field alignment within a given space

Associating with a Browse List

Enter the name of the browse list that you want to be associated with your field. To find the available browse lists, look under the Browser Library heading in the main Forms Design window. The following illustration shows a sample Browser Library listing.



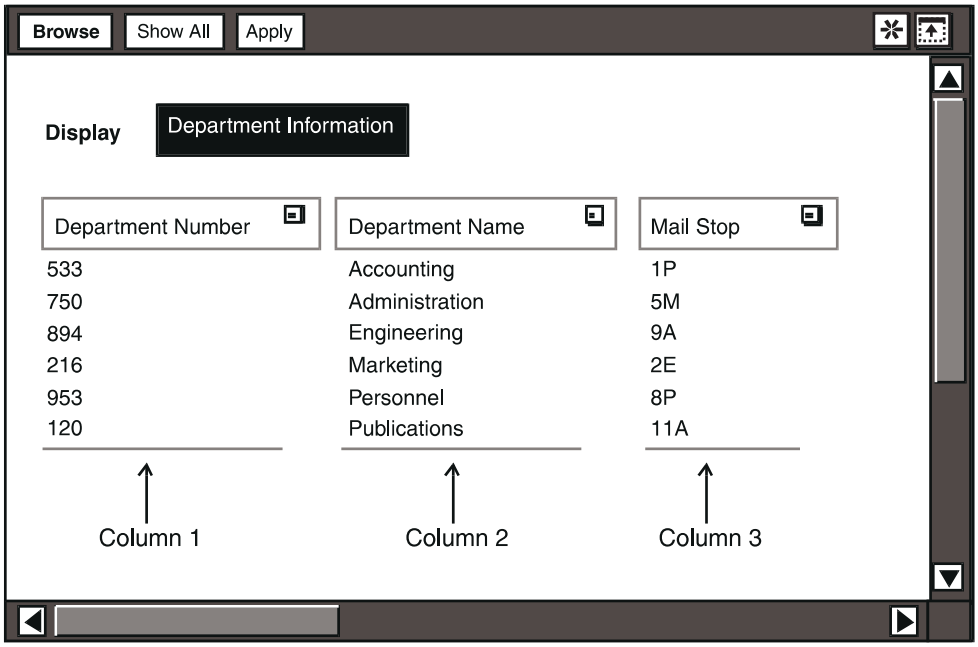
To establish a browse list for the **EMPLOYEE #** field shown in Figure 11 on page 36, choose the “Employee Information” browse list. Enter the name of this browse list into the **Browser Name** field.

Browser Name Employee Information

As forms designer, you determine the names and properties of the browse lists. For information about setting up browse lists, see Chapter 13, “Working with the Browse Feature” on page 90.

Figuring Out the Return Column

Within a browse list, there are usually several columns. For example, in a browse list for department information, there could be columns such as department number, department name, and mail stop. The browse window for such a browse list is shown in the following illustration.



The columns in a browse list are numbered consecutively from left to right, beginning with 1. In the above illustration, column 1 is Department Number, column 2 is Department Name, and column 3 is Mail Stop.

You specify which column's contents should be returned to your field by entering the appropriate column number in the **Return Column** field.

For example, if you enter **2** in the **Return Column** field, the browse list returns the contents from column 2, the **Department Name** column, when a selection is made from the browse list. The choices in the browse list can be constrained by using any column in the browse list, but a selection must be made from the return column in order for values to be entered in the field. If you do not specify a column, the **Return Column** field defaults to 1.

For more information on setting up and using browse lists in Forms Design, see Chapter 13, “Working with the Browse Feature” on page 90.

Displaying the Browse List

You can determine the behavior of the browse list with the **Browser Display** field.

The display mode for a browse list can be either manual or automatic. If you decide to set the **Browser Display** to **Manual**, the browse list appears only if the **Browse** button is clicked on. Use this method when the browse list is large, or when users are familiar

with the contents of the browse list. A user who knows what value needs to be entered does not want to wait for the browse list to appear.

If you want to encourage the user to select an entry through the browse list, set the **Browser Display** to **Auto** and the browse list appears automatically when the user selects in the field. The user can still enter information directly into the field, but it is easier to select the entry from the browse list. Use this method when the accuracy of the entry is imperative, or when users are less familiar with the contents of the browse list.

Entering the Browser Parameters

You can constrain the choices in the browse list by using this field in conjunction with the **General Qualifier** field or the **SQL For Update Value** field. For more information, see Chapter 13, “Working with the Browse Feature” on page 90.

Chapter 8. Designing Choice Fields

Choice fields offer you flexibility when designing forms by enabling you to create hierarchies of fields. The options discussed in this chapter appear in the field area of the field options windows when you select **User Choice** from the **Field Type** field. This chapter describes:

Presenting choices

Placing dependent fields

Setting the dimensions of choices

Defining choices

Setting dependencies between fields

See Chapter 6, “Discovering Data Entry Methods” on page 28, for a complete overview of the types of fields available to you. For information about options you can set in the other sections of the field options windows, see Chapter 11, “Initializing Fields” on page 76, and Chapter 12, “Storing Data” on page 85. For a field in a retrieval form, also see “Constraining Information Retrieval” on page 74.

Learning About Choice Fields

You define a field as a field when you want the user to select one of several possible choices. By creating a field, you reduce the number of keystrokes that must be entered, and also ensure that only valid values are entered.

A field is similar to a field that has a browse list. The main differences are:

The choices in a field appear in the Data Entry window.

The browse list for a field has too many entries to display in a field.

Therefore, if the list of possible values for a field is relatively small, and the entire list can be displayed in the Data Entry window, then the field can be defined as a field.

Creating Choice Fields

To create a field:

1. Select a field in one of the form depictions in the main Forms Design window.
2. Open the field options window.
3. Select **User Choice** in the **Field Type** field.

   The field area options for a field in an entry form appear.
4. Close the field options window.

© Copyright IBM Corp. 1996

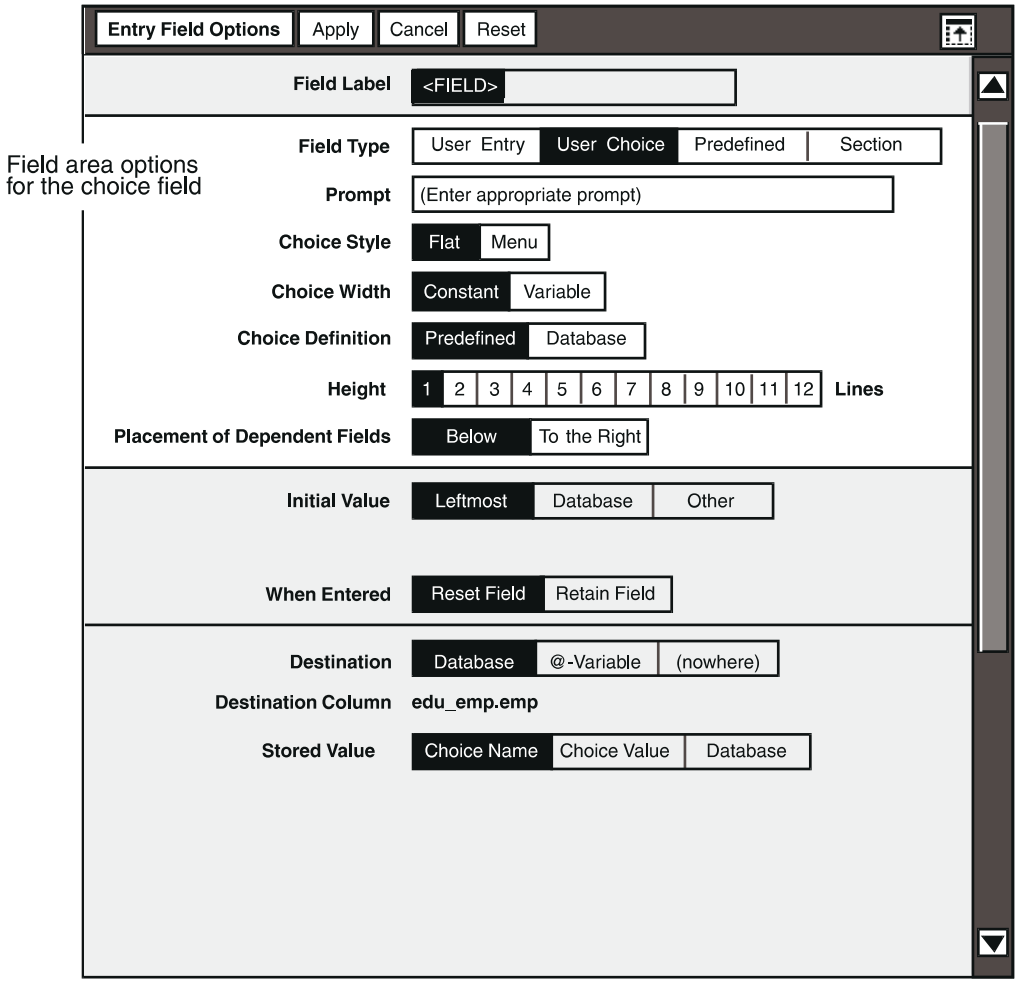


Figure 12. Field area options for a field

The field options for the field are discussed in the section “Setting Preferences for Choice Fields” on page 47.

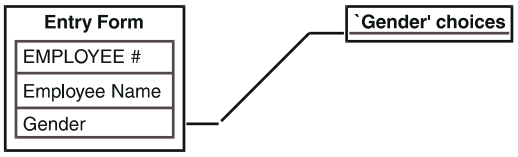
Adding Choice Field Choices

After you have defined a field as a field, you can create choices as shown in the following illustration.

To add choices to a field:

1. Double-click on the field in the form depiction. In this case, double-click on the **Gender** field in the **Entry Form** depiction.

   A heading for the 'Gender' choices appears. A line connects the choices heading to the field that it defines.



2. Select the choices heading.

   The choices heading becomes highlighted and the buttons in the main window header are replaced by a single **Add Choice** button.

3. Click on the **Add Choice** button.

   Each time you click on the **Add Choice** button, a field labeled <**CHOICE>** is added under the choices heading.

Naming Choices

After you have added choices to a field, you need to give them names that are meaningful to your users.

To name choices:

1. Select a choice within the depiction of your field in the Forms Design window.
2. Open the Choice Options window.
3. Enter a choice name in the **Choice Line 1** field.

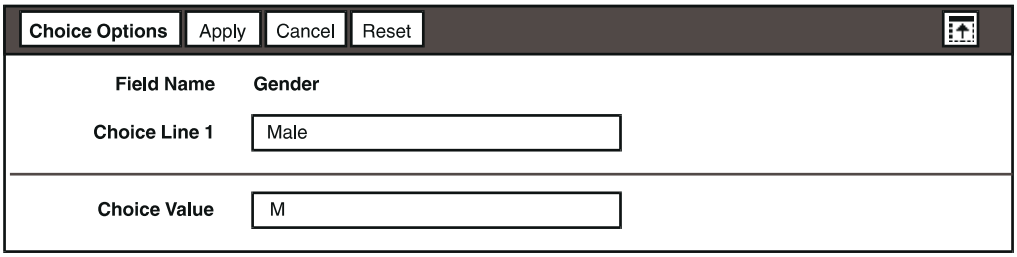
   This field contains the actual choice name that appears in the form. For example, if you enter **Male** in the **Choice Line 1** field, the **Gender** field contains a **Male** choice.

   If you set the **Height (Lines)** field to Choice Options window displays separate fields for each line in each choice.

4. Enter a value in the **Choice Value** field.

   When you predefine choices, you also redefine the choice value; the choice value is assigned a default number according to the order in which it was added. The choice value is a code associated with the choice.

   For example, the choice value for the **Male** choice can be changed from the default value of 1 to the letter M, as shown in the following illustration.

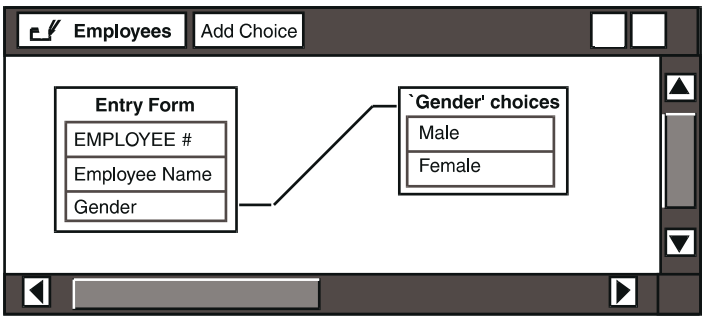


Although the user selects a choice name from a field, you can specify that either the choice name or the choice value be stored in the database. See Chapter 12, “Storing Data” on page 85, for more information on storing the choice name and choice value. See Chapter 11, “Initializing Fields” on page 76, for information on initializing a predefined choice from the database.

5. Close the Choice Options window.

   When the window closes, the name of the first field has changed to **Male**.

6. Repeat steps 1 through 5 to define any other choices. These steps were repeated to define the **Female** field shown in the following illustration.



7. Click on the original field in the form depiction to close the choices heading area. In this example, select the **Gender** field to close the 'Gender' choices heading area.

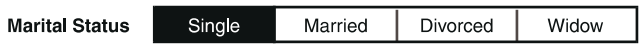
   The **Gender** field appears with the two choices in the Data Entry window, as shown in the following illustration.

Typically, the choice name, rather than the choice value, is stored in the database. However, if you want the choice value to be stored in the database, you have to specify additional information in the storage area of the field options window. See Chapter 12, “Storing Data” on page 85, for more information on storing the choice name and choice value.

Layering with Choice Fields

If you have a field for which you have defined choices, you can create dependent fields by defining fields under each choice. By creating dependent fields, you create hierarchies of fields and choices.

For example, the **Marital Status** field in an Employees table is a field that has been defined with several choices. The choices within the field are displayed in the following illustration.

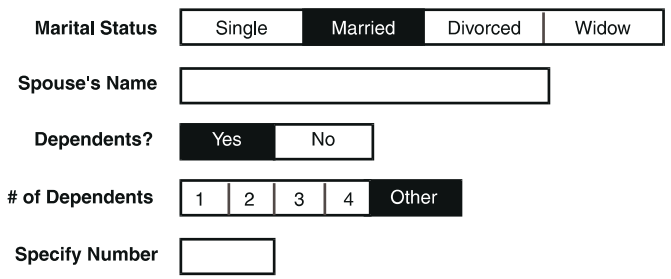


Each of the choices shown in the **Marital Status** field displays more fields. For example, if **Married** is selected then the **Spouse's Name** field appears, as shown in the following illustration. The **Spouse's Name** field is dependent on the selection of the **Married** choice.

You can also define choices to show other fields when selected. In other words, choices can have other choices that are dependent on them.

You can layer dependent fields as deep as necessary from the base predefined field to capture the information you need.

With a field, you can also provide the option to enter information rather than select one of the available choices. For example, the following illustration shows an additional dependent field, **# of Dependents**, that only appears when **Yes** is chosen from the **Dependents?** field. The **# of Dependents** field has a choice, **Other**, that allows the user to enter a number different than the choices given in the **# of Dependents** field.



When two fields in an entry form correspond to the same database column, the Data Entry tool stores the data from the field that is at the lowest level. In the following example, both the **# of Dependents** and **Specify Number** fields correspond to the same column in the database table.

If **1** is selected from the **# of Dependents** field, then the Data Entry tool stores the contents of the **# of Dependents** field in the database. However, if **Other** is selected

from the **# of Dependents** field, then the Data Entry tool stores the contents of the **Specify Number** field in the database.

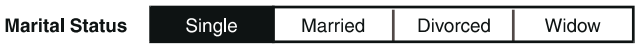
Creating Dependent Fields

To create dependencies between fields, you follow the same basic procedure no matter what type of dependent field you are creating. Further, you can create as many levels of dependent fields as you require to gather the necessary information. However, only choices that you create can have dependent fields; choices derived from database values cannot have dependent fields.

To create a dependent field:

1. Select a field in a form depiction in the main Forms Design window.

   Use the **Marital Status** field shown in the following illustration as the example. Select its depiction in the Forms Design window.

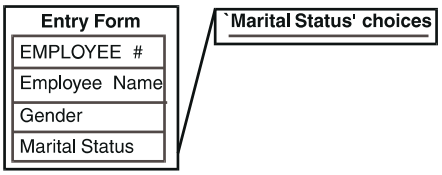


2. Open the field options window.
3. Select **Predefined** in the **Choice Definition** field.
4. Close the field options window.

   You have now specified that you are defining the choices for this field.

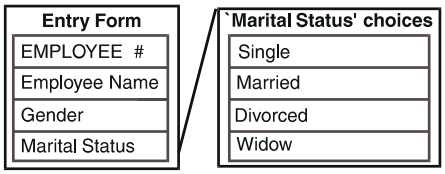
5. Double-click on the depiction of the field to display a choices heading.

   For example, if you double-click on the **Marital Status** field the 'Marital Status' choices heading appears as shown in the following illustration.



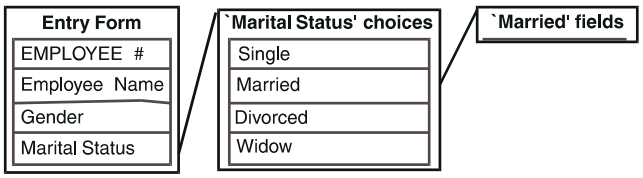
6. Add choices to the choice heading.

   For example, you could define **Single**, **Married**, **Divorced**, and **Widow** as the choices for the 'Marital Status' choices heading, as shown in the next illustration. See the section “Adding Choice Field Choices” on page 41 for more information on adding choices to a field.



7. Double-click on one of the choices that you have added to the choices heading.

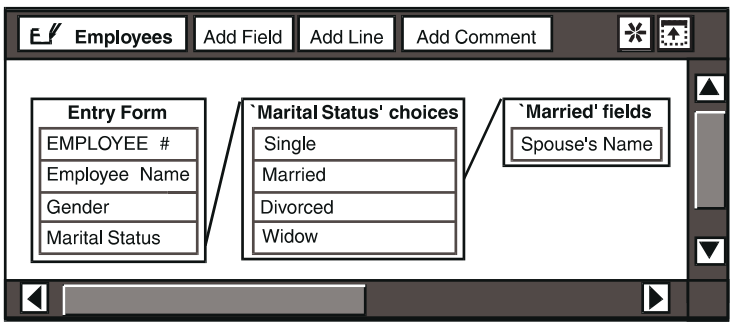
   A heading appears for the choice that you have selected. For example, if you double-click on **Married**, the 'Married' fields heading appears as shown in the following illustration.



   You can now define the dependent fields for the **Married** choice.

8. Add dependent fields to the fields heading.

   For example, add a field named **Spouse's Name** to the 'Married' fields heading.



9. Select the dependent field that you added to the heading, if it is not already selected.

   For example, select the **Spouse's Name** field shown in the previous illustration.

10. Open the field options window.

11. Define the dependent field as necessary.

    For example, to create the **Spouse's Name** field, you select **User Entry** in the **Field Type** field and type Spouse's Name.

12. Close the field options window.

The field now appears as a dependent field when the appropriate choice is selected.

For example, the **Spouse's Name** field appears as a field in the Data Entry icon when **Married** is selected from the **Marital Status** field, as shown in the following illustration.

Setting Preferences for Choice Fields

There are several options that you have when designing fields. You can set options to control how choices are defined and displayed in choice fields.

To set your preferences for a field:

1. Select a field in one of the form depictions in the main Forms Design window.
2. Open the field options window.
3. Select a choice for each field.
4. Close the form options window when complete.

Table 8 (Page 1 of 2). Choice field options

Field	Options	Function
Field Type	User Choice	Displays the options for a field.
Prompt	N/A	Prompts the user to perform an action, such as enter a value. The Forms Design tool initializes this field to the WTDD description of the column copied from the category window. For more information about entering a prompt message, see Chapter 6, “Discovering Data Entry Methods” on page 28.
Choice Style	Flat	Displays the choices within the field horizontally.
	Menu	Displays the choices within the field in a pull-down menu.
Choice Width	Constant	Makes each choice large enough to contain the widest choice.
	Variable	Makes each choice width differ according to the length of the choice.

Table 8 (Page 2 of 2). Choice field options

Field	Options	Function
Choice Definition	Predefined	You determine the choices for the field.
	Database	When you select **Predefined** in the **Choice Definition** field, the **Height (Lines)** field and **Placement of Dependent Fields** fields appear.
		The database generates the choices according to the **SQL For Choices** field.
		When you select **Database** in the **Choice Definition** field, the **SQL For Choices** field appears.
Height (Lines)	1 to 12	Sets the number of lines of text that appear for each choice.
SQL For Choices	N/A	Enters the SQL select statement that generates the choices.
Placement of Dependent Fields	Below	Places the dependent fields for a choice below the field.
	To the Right	Places the dependent fields for a choice next to the field.

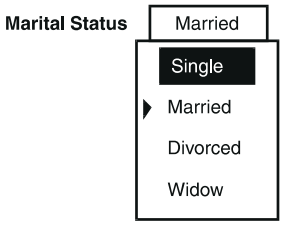
Presenting Choices

You determine the presentation style of choices within a field. The choices can be presented horizontally or in a pull-down menu.

If you select **Flat** as the presentation style, the field appears as shown in the following illustration.

If you select **Menu** as the presentation style, the field appears as shown in the following illustration.

When the user selects the visible choice, a pull-down menu with several other choices appears. The menu only appears when the user selects on the visible choice. The user can change the visible choice by moving the cursor down to highlight the desired choice as shown in the next illustration.



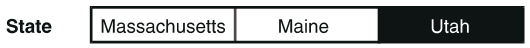
Notice the arrow to the left of the Married choice. The arrow points to whichever choice is the current visible selection within the field. When a different selection is made, the field reappears showing only the new choice, as shown in the following illustration.

The menu style is useful if you have several choices for your field, and you want to conserve space on your form by having the other choices display only when the visible choice is selected.

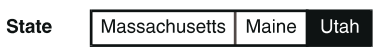
Setting the Choice Width

You determine the width of the choices in your field. You can make each choice a standard width, or you can vary the width to suit the length of each choice.

To set a standard display width for each choice, select **Constant** in the **Choice Width** field. The display width of every choice is the same, and the width is equal to the width required to display the longest choice name in the field. The next example shows a field with a constant choice width.



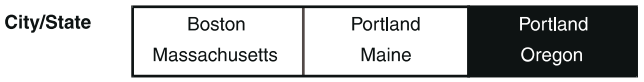
To vary the width of each choice, select **Variable** in the **Choice Width** field. When you have many choices, selecting **Variable** helps save space. The display width of each choice is determined by the length of the particular choice name. The next example shows a field with a variable choice width.



Setting the Choice Height

You determine the height of the choices in your field. You can determine the height by the number of lines within the **Height (Lines)** field.

For example, if each choice consists of a city and state, you select **2** in the **Height (Lines)** field so that each choice appears in two lines.



Deciding How to Define Choices

You select a choice in the **Choice Definition** field to determine the data source for a field. The choices in the **Choice Definition** field create fields with data that are either:

Defined by you, the forms designer

Generated dynamically from the database server

These data sources create two types of fields and are described in the following sections.

Defining the Choices Yourself

When creating the form, the **Predefined** choice allows you to specify the exact choices from which a user can select. This method of defining choices is described in the earlier section “Adding Choice Field Choices” on page 41.

Generating the Choices Dynamically

The **Database** choice in the **Choice Definition** field of the field options window lets you request that the database generate field values. The database then provides the choices from which a user selects.

For choices that are database-defined, the choice value assigned to each choice is its sequence number within the field. Choices are numbered from left to right, or top to bottom, beginning with 1 (one).

To generate choices from the database for a field:

1. Copy a field from the category window to a form depiction in the main Forms Design window. As an example, consider the Employees and Departments tables shown in Figure 13 on page 51.

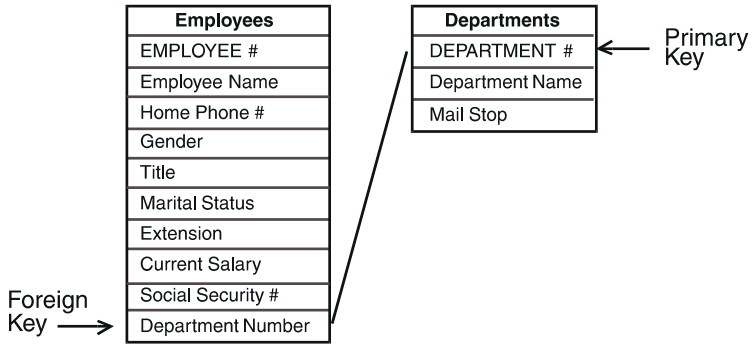


Figure 13. Foreign and primary keys in the category window

There are two types of fields you can copy from the category window:

Foreign key field

The **Department Number** field in the Employees table is a foreign key; it references the primary key (**DEPARTMENT #**) in the **Departments** table. The choices for this field should be the contents of the DEPARTMENT # column in the Departments table.

Primary key field

The **DEPARTMENT #** field in the Departments table is a primary key for the Departments table.

If the field that you copy from the category window is a foreign key, you might want the choices that appear in the field to be generated from information stored in the table that defines the key values. This method ensures a faster response time and generates distinct values for the choice field.

For example, if you copy the **Department Number** field from the Employees table into an Entry Form depiction, you can define that field as a field and specify that the choices for that field should be the primary key values in the Departments table.

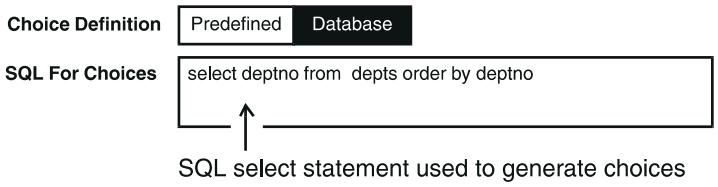
2. Open the field options window.
3. Select **User Choice** in the **Field Type** field.
4. Select **Database** in the **Choice Definition** field.

   The **SQL For Choices** field appears.
5. Enter an SQL statement in the **SQL For Choices** field.

   When a field is database-defined, you must provide an SQL statement that the Data Entry tool executes to generate the choices for the field. The SQL should generate distinct choices so that there are no duplicate values. The Data Entry tool executes this statement when the icon is opened.

   The choices generated by the example shown in the following illustration contain all the department numbers that are defined in the dimension table. This information

is requested in the **SQL For Choices** field, where the database table to be used is named.



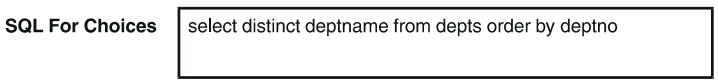
The **SQL For Choices** field appears only when **Database** is selected from the **Choice Definition** field. When you select **Predefined**, the **Height (Lines)** field and **Placement of Dependent Fields** fields appear instead of the **SQL For Choices** field.

You can use @-keywords in the SQL For Choices statement. See “Discovering the @-Keywords” on page 145 for information about the @-keywords.

6. Enter a new SQL statement in the **SQL For Choices** field if you want to further define the choices to be used for the choice field.

   Be sure to constrain the SQL statement sufficiently so that an appropriate number of choices is returned from the database.

   In this example, you can further define the choices to contain the department names, instead of the department numbers. Notice that the SQL statement in the **SQL For Choices** field, in the following illustration, is slightly different from the one shown in the previous illustration. The term *distinct* is included in the SQL statement to guarantee that no duplicate values are returned from the database.



   When you define a field to contain department names and you want to store the department numbers in the database, you must specify additional information in the storage area of the field options window. See Chapter 12, “Storing Data” on page 85, for more information.

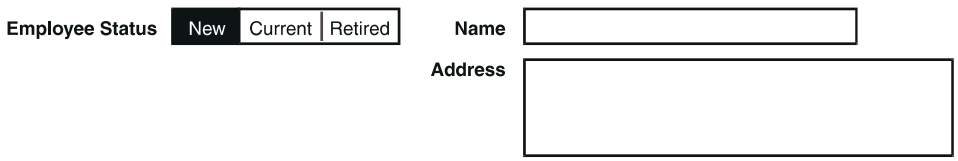
7. Close the field options window when complete.

Positioning Dependent Fields

You can determine the placement of dependent fields for a choice in relation to the field.

You can display the dependent fields below or to the right of the choice field.

To display dependent fields to the right of the field, select **To the Right** in the **Placement of Dependent Fields** field. In this case, the **Name** and **Address** fields appear as shown in the following illustration:

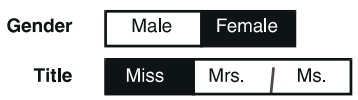


Notice that when you place the dependent fields to the right of the choice field, they are placed as a block. If you want the dependent fields placed horizontal to one another and to the field, you need to specify a group for the dependent fields. See Chapter 14, “Enhancing Forms with Design Techniques” on page 106, for more information about groups. See the section “Creating Dependent Fields” on page 45 for more information about defining dependent fields for choices.

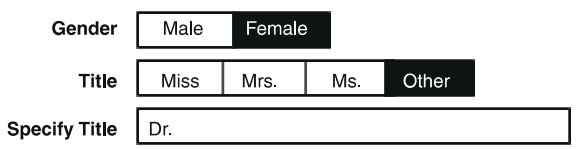
Enhancing Retrieval Form Choice Fields

When you create a retrieval form that contains choices, you should be careful to create fields that allow retrieval of the necessary data.

For example, if you use the choices shown in the following illustration for a **Title** field in a retrieval form, the user is only able to retrieve rows for which the **Title** field is set to Miss, Mrs., or Ms.



This construction excludes rows for which the **Title** field might be Dr. To enable users to retrieve other titles, you should include **Other** in the choice field. When the user selects **Other**, a **Specify Title** field appears, and the user can then specify a title.



Chapter 9. Building Flexibility with the Section Field

Section fields are useful when you need to break information into distinct segments, and display only one segment of information at a time. The options discussed in this chapter appear in the field area of the field options window when you select the **Section** choice in the **Field Type** field. This chapter describes:

Creating section fields

Defining dependent fields for section choices

Presenting section choices

Placing dependent fields

Setting the height and width of section choices

See Chapter 6, “Discovering Data Entry Methods” on page 28, for a complete overview of the types of fields available to you. For information about options you can set in the other sections of the field options windows, see Chapter 11, “Initializing Fields” on page 76, and Chapter 12, “Storing Data” on page 85. For section fields within the retrieval form, also see “Constraining Information Retrieval” on page 74.

Learning About Section Fields

Define a section field when the information in the form you are creating can be organized into independent segments. Also, unlike fields, a user can make selections across the sections of a section field without losing the information, and without entering the selections until finished. Entries to the entire section field are validated when the user completes data entry and clicks on the **Enter** button.

A section field is similar in structure and operation to a field that has been predefined. The main features of the section field are:

The application process

Use the section field when your form contains multiple sections that must be completed before validating the data.

The validation process

Use the section field when the validation process must involve all dependent fields within the section field, not only those displayed under the current selection.

The data entry process

Use the section field when you want the user to complete all sections before clicking on the **Enter** button.

© Copyright IBM Corp. 1996

Creating Section Fields

To create a section field:

1. Select a field in one of the form depictions in the main Forms Design window.
2. Open the field options window.
3. Select **Section** in the **Field Type** field.

   The field options window for the section field appears.

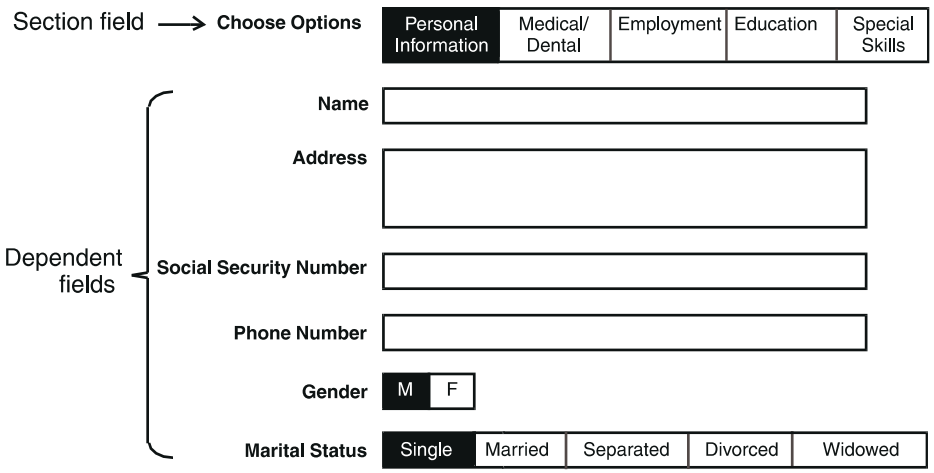


Figure 14. Field area options for a section field

The field options for the section field are discussed in the section “Setting Preferences for Section Fields” on page 58.

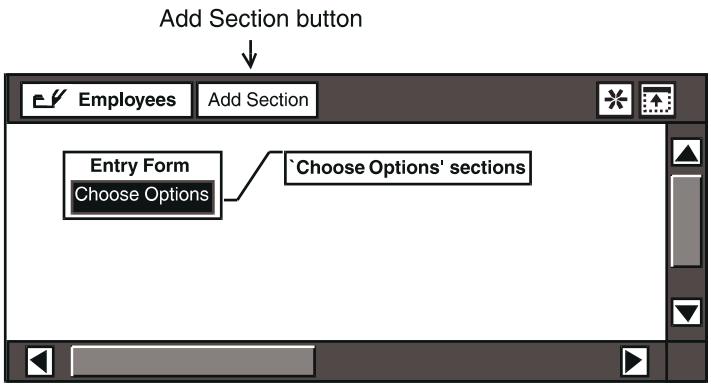
Adding Section Field Choices

After you have defined a field as a section field in the field options window, you can create section choices.

To create choices in a section field:

1. Double-click on the section field in the form depiction in the main Forms Design window.

   A section heading appears with a line that connects it to the field that it defines. In the case of the **Choose Options** section field, the heading reads 'Choose Options' sections.



2. Select the section heading.

   The section heading becomes highlighted and the buttons in the header of the main window are replaced by a single **Add Section** button.

3. Click on the **Add Section** button.

   A field labeled <**SECTION>** is added under the section heading. Each field you create represents a choice within the section field.

4. Select a choice under the section heading.
5. Open the Section Options window.

   If you do not specify the **Height** to be two lines in the field options window, then only one **Section Line** field appears.

6. Enter a name in the **Section Line** fields.

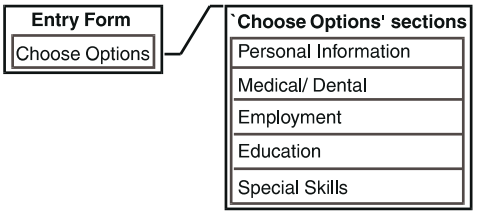
   The name entered in the **Section Line** fields is the section choice that appears in a data entry form. If the section choices contain multiple lines, a **Section Line** field is displayed for each line in the section choice.

7. Enter data in the **Section Value** field.

   The section value is a code associated with the section choice. Although the user selects a section name, you can specify that either the name or the value be stored in the database.

8. Close the Section Options window.
9. Repeat steps 4 through 8 for each section choice that you want to create within the section field.

After you create and define all the section choices for the **Choose Options** section field, the depiction of the section appears as shown in the following illustration.



Organizing Information into Sections

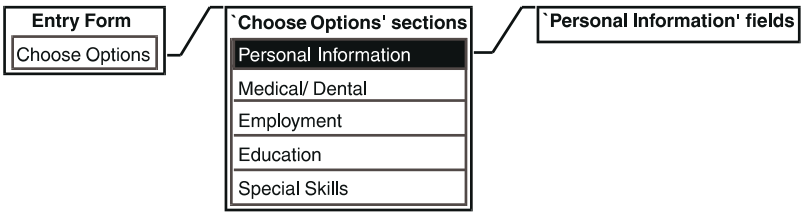
If you have a section field, you can organize information into appropriate sections. Under each section choice, create hierarchies of fields and choices.

In the following example, the **Choose Options** section field is used.

To create a field that depends on a section choice:

1. Double-click on one of the section choices to display a section fields heading.

   For example, if you double-click on the **Personal Information** section choice, the 'Personal Information' fields heading appears as shown in the following illustration.



   You can now define the dependent fields for the **Personal Information** section choice.

2. Select the section fields heading. The **Add Field**, **Add Line** and **Add Comment** buttons are restored in the window header.
3. You can either copy a field from the category window to the new field's heading, or use the **Add Field** button.

   To create a **Name** field, add a field to the new field's heading with the **Add Field** button. Open the field options window for the new field and select **Entry** in the **Field Type** field. Enter *Name* as the field label.

You can create dependent fields of any field type, and even embed section fields within one another. Experiment with the flexibility of the Forms Design tool. The layering technique was designed to save you form space.

Setting Preferences for Section Fields

To set your preferences for the section field:

1. Select a section field in one of the form depictions in the main Forms Design window.
2. Open the field options window.

   The field options window for the section field appears.
3. Select a choice for each field.

   All of the options in the field type area are described in Table 9. A section field is used to enter data in an entry form, update data in an update form, and constrain data in a retrieval form.
4. Close the field options window when complete.

Table 9. Section field options

Field	Options	Function
Field Type	Section	Displays the options for a section field.
Prompt	N/A	Prompts the user to perform an action, such as enter a value. The Forms Design tool initializes this field to the WTDD description of the column copied from the category window. See Chapter 6, “Discovering Data Entry Methods” on page 28,” for more information about entering a prompt message.
Choice Style	Flat	Displays the choices within the field horizontally.
	Menu	Displays the choices within the field in a pull-down menu.
Placement of Dependent Fields	Below	Places the dependent fields for a choice below the section field.
	To the Right	Places the dependent fields for a choice to the right of the section field.
Choice Width	Constant	Makes each choice width large enough to contain the longest choice name.
	Variable	Makes each choice width differ according to the length of each choice name.
Height (Lines)	1 to 12	Sets the number of lines of text that appear for each choice.

Presenting the Choices

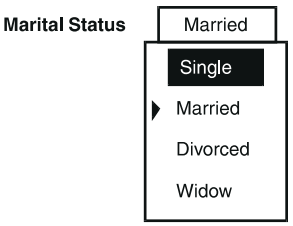
You determine the presentation style of choices within a section field. The choices can be presented horizontally or in a pull-down menu.

Choice Style Flat | Menu

If you select **Flat** as the presentation style, the field appears as shown in the following illustration.

If you select **Menu** as the presentation style, the field appears as shown in the following illustration.

When the user selects the visible choice, a pull-down menu with several other choices appears. The menu only appears when the user selects the visible choice. The user can change the visible choice by moving the cursor down to highlight the desired choice as shown in the next illustration.



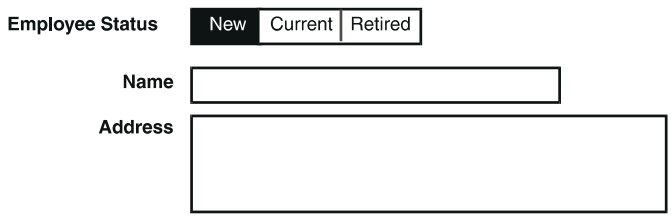
Notice the arrow to the left of the **Married** choice. The arrow points to whichever choice is the current visible selection within the field. When a different selection is made, the field reappears showing only the new choice, as shown in the following illustration.

The menu style is useful if you have several choices for your field, and you want to conserve space on your form by having the other choices display only when the visible choice is selected.

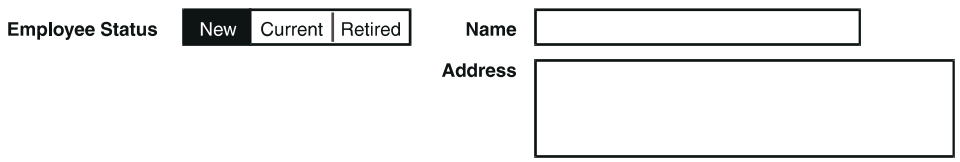
Positioning Dependent Fields

You can determine the placement of dependent fields for a choice in relation to the section field.

The following illustration shows a section field, **Employee Status**, with dependent fields for the **New** choice placed below it.



You can also display the dependent fields to the right of the section field by selecting **To the Right** in the **Placement of Dependent Fields** field. In this case, the **Name** and **Address** fields appear as shown in the following illustration.

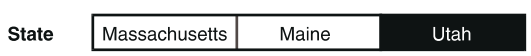


Notice that when you place the dependent fields to the right of the section field, they are placed as a block. If you want the dependent fields placed horizontal to one another and to the section field, you need to specify a group for the dependent fields. See Chapter 14, “Enhancing Forms with Design Techniques” on page 106, for information about groups. See the section “Organizing Information into Sections” on page 57 in this chapter for information about defining dependent fields.

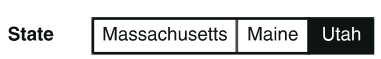
Setting the Choice Width

You determine the width of the choices in your field. You can make each choice a standard width, or you can vary the width to suit the length of each choice.

To set a standard display width for each choice, select **Constant** in the **Choice Width** field. The display width of every choice is the same, and the width is equal to the width required to display the longest choice name in the section field. The next example shows a section field with a constant choice width.



To vary the width of each choice, select **Variable** in the **Choice Width** field. When you have many choices, selecting **Variable** helps save space. The display width of each choice is determined by the length of the particular choice name. The next example shows a section field with a variable choice width.



Setting the Height of a Section Field

You can determine the height of a section field by selecting a choice in the **Height (Lines)** field.

The height of a section field is measured relatively by lines of text.

For example, the following section field has two lines specified in the **Height (Lines)** field.

Chapter 10. Validating Data

You can set several options to control how the Data Entry tool validates data in fields. You can also set the editing options that are available for fields containing character strings. The majority of the options described in this chapter appear in the validation area of the field options window when **User Entry** is selected. Validation options for choice and section fields in the retrieval form are also discussed. This chapter describes:

Entry options for the user

Acceptable data types

Valid database values

Validation methods

Default validation information

Validation of choice and section field entries for the retrieval form

Discovering Validation Options

You can set the following validation options for fields:

Entry must be single

Value must be an acceptable data type

Value must be within a range of values

Character string must match a pattern

Value must be a valid database value

Value must be in a browse list

To display the field options window:

1. Select a field in a form depiction in the main Forms Design window.
2. Open the field options window. This window looks similar to the Entry Field Options window shown in Figure 15 on page 63.

   The options shown in the validation section of the Entry Field Options window are the same as the validation options in the Retrieval Field Options and Update Field Options windows for a field.

© Copyright IBM Corp. 1996

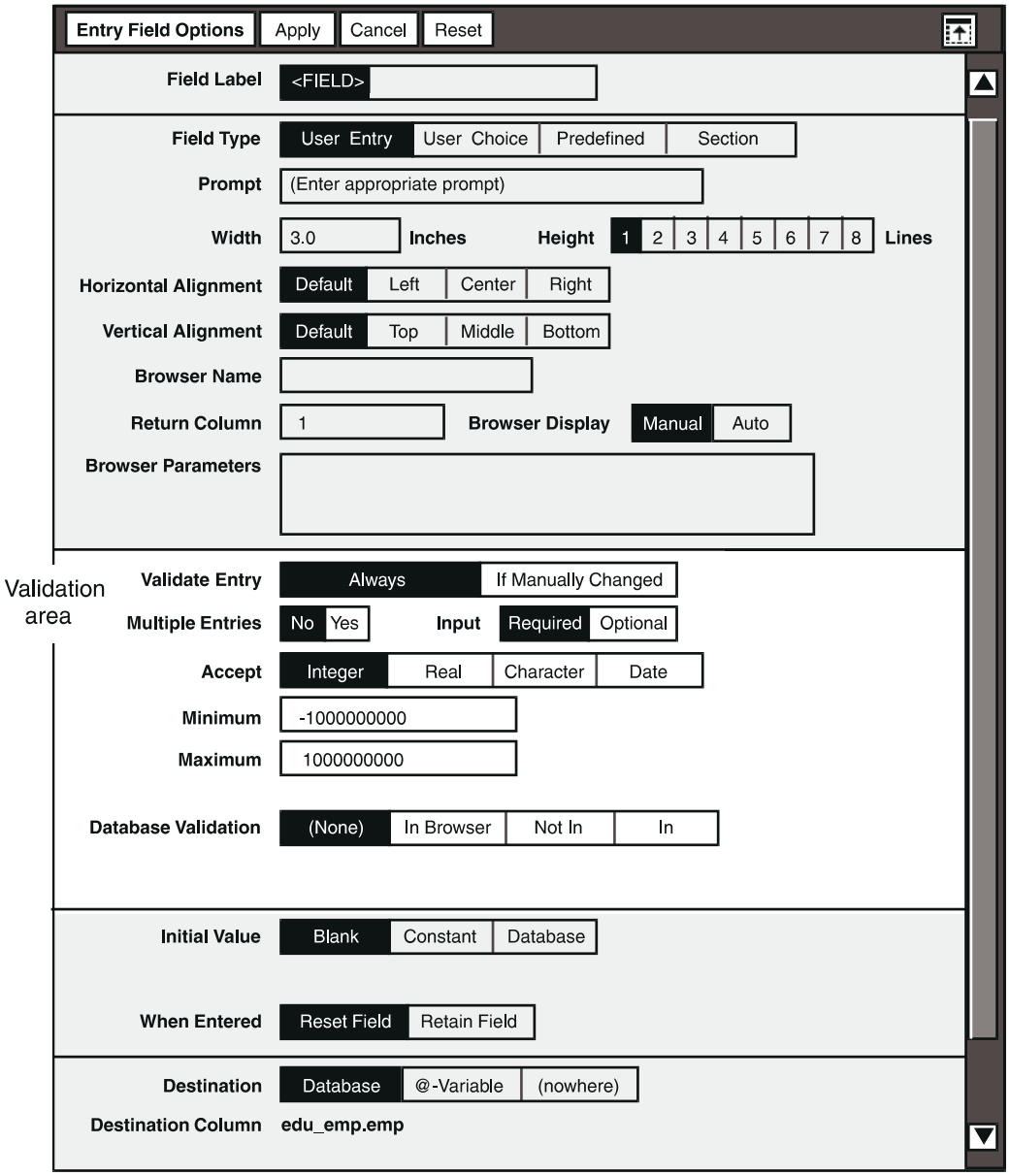


Figure 15. Validation area in the Entry Field Options window

The Data Entry tool applies validation options to entries in the order in which the options appear in the field options window. For example, the Data Entry tool first verifies whether the entry should be validated or not as directed by the **Validate Entry** field. Then, if validation is necessary, the Data Entry tool verifies whether multiple

entries are allowed in the field. This procedure continues until all of the specifications set in the validation area are verified against the entry.

Determining the Validation Method

You determine when an entry within a field is validated by selecting a choice from the **Validate Entry** field. This field defaults to the Always choice.

Always Entries are validated each time the user clicks on the **Enter** button. Use this method for dynamic databases where information is constantly changing.

If Manually Changed

Entries are validated only when manually changed, then entered. Use this method when the information in the database is fairly static.

The field is considered manually changed when the user types the entry directly from the keyboard or changes an existing entry.

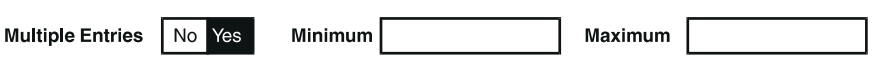
The field is not considered manually changed and, consequently, should not be validated when the entry is deleted completely, the entry is selected from the browse list, or the entry has already been validated.

Specifying Single or Multiple Entries in an Entry Field

You can limit a field to one data item, or allow several data items, by using the **Multiple Entries** field.

When you select **No** in the **Multiple Entries** field, the Data Entry form assumes only one value will be entered into the field. Therefore, commas are interpreted as part of a string rather than as a separator between entries. For example, the entry **IBM, IBM** is interpreted as one entry rather than two separate entries. Select **No** when the user is entering strings that include a comma.

To instruct the Data Entry form to look for multiple entries within a field, select **Yes** in the **Multiple Entries** field. When you select **Yes**, the **Minimum** and **Maximum** fields appear as shown in the following illustration. These fields let you specify how many items are accepted within the field.



The **Multiple Entries** field is supported only by the entry and update forms. Further, you cannot allow multiple entries when entering data into a database.

The following conditions apply to multiple entries within a field:

The **Minimum** parameter must be an integer greater than or equal to 0. If 0 is entered, then input is not required for the field; if a value greater than or equal to 1 is entered, then input is required.

The **Maximum** parameter must be an integer greater than or equal to the minimum number.

The **Minimum** and **Maximum** field values can be a maximum of 255.

The data items must be separated by a comma and a space within the entry field. If a data item already contains a comma and a space, the entire item must be enclosed within single quotes. For example, **'Miami, Florida'**.

Each data item within the field is validated individually.

If you want the Data Entry form to accept only one entry, and to post an error message at multiple entries, then select **Yes** in the **Multiple Entries** field. Then set both the **Minimum** and **Maximum** fields to **1** to ensure only one entry.

Making Input Required or Optional

In the **Input** field, you determine whether or not input is required in the field.

Select **Required** if the user must enter a value in the field. If you select **Optional**, no entries are necessary in the field.

When **Multiple Entries** is set to **Yes**, the **Minimum** and **Maximum** fields replace the **Input** field. You can still make input optional by setting the minimum to 0.

If you select **Optional** in the **Input** field for an integer, real, or date field, and the user leaves a blank field, the Data Entry tool checks the Workstation Tools Data Dictionary (WTDD) to see if an N/A value is defined for that column. If the value is defined, the Data Entry tool enters that value into the database column. If no N/A value is defined, that column is not specified in the insert command. On most database systems, the lack of a defined N/A value results in a null value in the database.

Specifying an Acceptable Data Type Value

You specify that a field must contain an integer, real number, character, or date by selecting the appropriate choice in the **Accept** field.

Integer

Whole numbers for which you specify a maximum and a minimum value.

Real

Numbers that can contain a decimal point for which you specify a maximum and a minimum value. See the section “Designing an Entry Form” on page 13 for information about how to set the default precision for fields that contain real numbers.

Character

Strings of alphanumeric characters for which you specify the maximum length. The maximum length of a character string is 32,000.

Date

Dates that you can format by day, month, year, or bimonth. You can also specify that the date be within a maximum and a minimum period.

The following sections explain these choices in greater detail.

Creating Integer Value Ranges

Set the range for integer entries by entering high and low values in the **Minimum** and **Maximum** fields. Database validation is described later in this chapter.

To set up the validation options for a field that contains an integer:

1. Select **Integer** in the **Accept** field.

   Default **Minimum** and **Maximum** values are displayed.

2. Enter a new value into the **Minimum** field.

   For example, if you want to create a range of values for an **Employee Number** field, set the field to accept a minimum value of **1**.

3. Enter a new value into the **Maximum** field.

   Using the same example as in step 2, you could set a top range for the **Employee Number** field to accept a maximum of **500**.

Creating Real Number Value Ranges

Set the range of real number entries by entering high and low values in the **Minimum** and **Maximum** fields.

To set up the validation options for a field that contains a real number:

1. Select **Real** in the **Accept** field.

   Default **Minimum** and **Maximum** values are displayed.

2. Enter a value into the **Minimum** field.

   For example, if you want to create a range of values for a monthly salary field, set the field to accept a minimum value of **1000.00**.

3. Enter a value into the **Maximum** field.

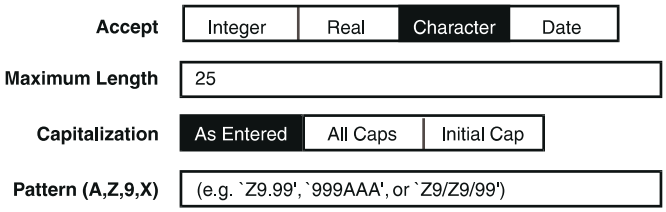
   Using the same example in step 2, you could set a top range for the monthly salary field to accept a maximum of **7999.99**.

Setting Character String Restrictions

Specify that a character string entry must be a certain length, capitalization style, and pattern.

To set up the validation options for a field that contains a character string:

1. Select **Character** in the **Accept** field.



2. Enter a value into the **Maximum Length** field.

   Determine the maximum number of characters that can be entered. For example, to set up validation options for a field containing Social Security numbers enter **11** in the **Maximum Length** field. You need a total of 11 spaces because there are 9 digits and 2 hyphens in the Social Security number. The maximum length allowed for a character string is 32,000. If the user enters more characters than you specify as the maximum length, an error message appears.

3. Select a choice from the **Capitalization** field.

   Decide how the Data Entry tool edits field entries. If you select **As Entered**, no changes are made to the entered character string. A selection of **All Caps** means that all characters are automatically capitalized. Select **Initial Cap** to automatically capitalize only the first character.

4. Enter a pattern into the **Pattern** field.

   The value that is entered into the field must match the pattern that you create. If the pattern entered does not match the pattern you specify, an error message appears.

   For example, to create a pattern for Social Security numbers, enter **999-99-9999** into the **Pattern** field. The hyphens are imposed in the pattern so that the Social Security number must be entered in the specified format. The **Pattern** field appears as shown in the next illustration.

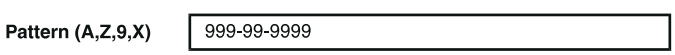


Table 10 defines the acceptable character formats that you can use in a pattern.

Table 10 (Page 1 of 2). Pattern characters

Pattern characters	Acceptable characters
A	a-z, A-Z
A	A
9	0-9
9	9
Z	0-9, space, null
Z	Z

Table 10 (Page 2 of 2). Pattern characters

Pattern characters	Acceptable characters
X	any character
X	X

Table 11 contains several examples of patterns, and shows acceptable and unacceptable data for each pattern.

Table 11. Pattern examples

Pattern	Acceptable data	Unacceptable data
999-99-9999	103-32-8172	2494-44-4321
AA-999	AB-121	A2-352
	aa-502	AA-59a
\AA-999	AZ-352	aC-123
aa-999	aa-123	AA-123
XZZ	@55	AB
	@ 5	B1A
Z9:99	1:23	:23
	12:34	1:2

The following conditions apply when creating patterns for a field:

A backslash symbol (\) indicates that only the character directly following it is acceptable. As an example, if the pattern in the **Pattern** field is **A**, then the only character allowed is **A** (this pattern is case-sensitive, so **a** is unacceptable). However, if the pattern is **A** (without the backslash), then any character from **A** to **Z** is acceptable (including lowercase).

Other characters, such as the hyphen in a social security number, can be imposed in a specific position within a pattern indicating that only that character can be entered.

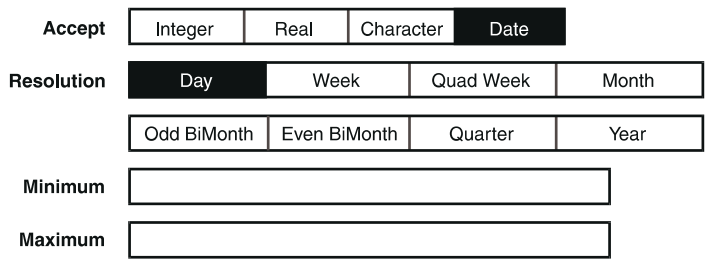
Pattern characters cannot be limited to a specified group, such as **A** to **F**.

Creating Date Value Ranges

Determine that the values for a date must be within a certain range by setting a precision for the date, and entering values in the **Minimum** and **Maximum** fields.

To set up the validation options for a field that contains a date:

1. Select **Date** in the **Accept** field. Further descriptive date options are displayed as shown in the following illustration.



2. Select a choice in the **Resolution** field.

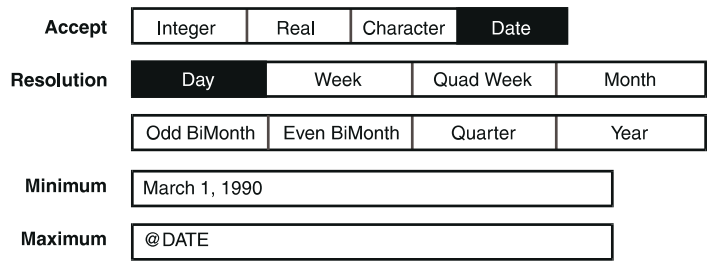
   Break down the date into the nearest day, month, bimonth, or year. For example, select **Day** if you want information specific to certain days.

3. Enter a value into the **Minimum** field that coincides with your choice in the **Resolution** field.

   For example, if you select **Day** in the **Resolution** field, enter a specific day into the **Minimum** field. You enter a date such as **March 1, 1990**.

4. Enter a value into the **Maximum** field.

   Enter a date in the same format as you did for the **Minimum** field. You also have the option of entering **@DATE.** The @-keyword @DATE represents the current date: the day that the user enters data into the form. In this case the options are displayed as shown in the following illustration.



This example shows a date field resolved to the nearest day, with a value range between March 1, 1990 and the present date.

Table 12 on page 70 lists the acceptable formats for entering dates.

Table 12 (Page 1 of 2). Acceptable date formats for direct entry

Date type	Correct formats	Incorrect formats
Day	January 15, 1994	Jan. 15, 1994
	January, 15, 1994	1-15-94
	January.15.94	
	Jan 15, 1994	
	jan 15, 94	
	jan 5 94	
	Jan/15/94	
	15/January/94	
	94.January.15	
	1/15/94	
	15.1.94	
	94-1-15	
	this day	
	today	
	Jan 15, 94 + 2 days	
Week	jan 15, 94 - 2 weeks	
	this week	
	week of Jan 15, 94	
	15 w 94	
	week of 1/15/94	
Quad Week	quad week of Jan 15, 94	
Month	January, 1994	Jan., 1994
	Jan, 1994	jan, 1994
	january, 94	January 1-94
	jan, 94	
	jan 94	
	this month	
	1/94 + 2 months	
Odd BiMonth	JF, 1994	Jan-Feb, 1994
	jf 94	
	1994 JF	
Even BiMonth	JF, 94	Jan-Feb, 94
	jf 1994	
	1994 JF	

Table 12 (Page 2 of 2). Acceptable date formats for direct entry

Date type	Correct formats	Incorrect formats
Quarter	1q94	First Quarter of 1994
	1Q94	
	this quarter	
Year	1994	94
	94	
	this year	

Specifying Database Validation

You can specify how the entered value must be validated by using the **Database Validation** field. Use this field with each of the data types that you specify in the **Accept** field.

The following sections of this chapter describe each of the four choices in the **Database Validation** field:

(None)

The value entered does not require database validation. Any value entered that meets the minimum and maximum criteria is accepted.

In Browser

A list must be generated from an applicable database table. The user browses this list and selects a valid value. For example, present a browse list if the entries consist of lengthy product names. See Chapter 13, “Working with the Browse Feature” on page 90, for more information.

Not In

The value entered should not exist in a database table. Use this choice if the entry must be a unique value; for example, an invoice order number.

In

The value entered must exist in a database table. Use this choice if the entry is in a format that is commonly understood, such as a date. Supplying a browse list of dates takes extra time to display, and uses unnecessary desktop disk space and random access memory (RAM).

The Not In and In choices require an SQL statement.

Setting No Database Validation

You can create a field that allows the user to enter any value, as long as the data type is correct. The data is not validated against a database table.

To create a field that allows any value:

1. Select **(None)** in the **Database Validation** field.

The **Database Validation** field opens.

Validating Data with a Browse List

You can create a field that allows the user to enter or select only those values that exist in a browse list.

To create a field that validates with a browse list:

1. Select **In Browser** in the **Database Validation** field.

The **Database Validation** field opens.

Browse lists are usually updated only when requested. However, depending upon how frequently the database changes, you can specify that the lists be updated each time the icon is opened, each time the lists are needed for validation, or based on a specified SQL statement. For more information about browse lists, see Chapter 13, “Working with the Browse Feature” on page 90.

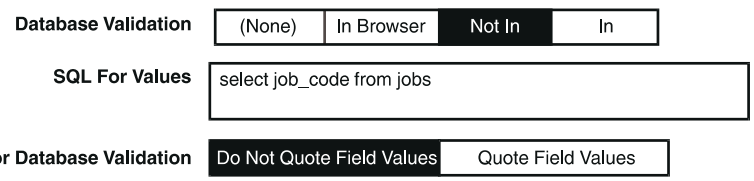
Preventing Duplicated Data

You can ensure that entries are not duplicated by specifying that the value entered is not currently in a database table. This type of validation is useful for fields that contain primary keys.

To ensure that entries do not exist in a database table:

1. Select **Not In** in the **Database Validation** field.

   Two additional fields are displayed along with the **Database Validation** field as shown in the following illustration.



2. Enter an SQL statement in the **SQL For Values** field.

   The statement in the previous illustration verifies that the entered value does not match any current job_code values in the jobs table. See the section “Entering an SQL Statement” on page 73 for more information about SQL.

3. Select a choice from the **For Database Validation** field.

   Choose **Do Not Quote Field Values** if the entry should be validated as entered, such as for numeric field values.

Choose **Quote Field Values** if the entry should be validated as a character string and enclosed in single quotation marks. Always use **Quote Field Values** for character string field values.

Validating Data in a Database Table

You can specify that the entry must exist in a database table. This type of validation is useful for fields that contain foreign keys.

To validate entries against a database table:

1. Select **In** in the **Database Validation** field.

   Two additional fields are displayed along with the **Database Validation** field.

2. Enter an SQL statement in the **SQL For Values** field.

   In this example, entries are validated against the jobs table. See "Entering an SQL Statement" for information about SQL.

3. Select a choice from the **For Database Validation** field.

   Choose **Do Not Quote Field Values** if the entry should be validated as entered, such as for numeric field values.

   Choose **Quote Field Values** if the entry should be validated as a character string and enclosed in single quotation marks. Always use **Quote Field Values** for character string field values.

Entering an SQL Statement

When you select **Not In** or **In**, the **SQL For Values** field appears below the **Database Validation** field.

The statement that you enter in an **SQL For Values** field generates the database values used for validation. For example, `select emp_no from Personnel`.

The &CONSTRAINT function is useful for constructing an SQL constraint for a field. See "Inserting or Updating Data in More Than One Table" on page 121 for more information.

You can also use @-keywords in the SQL statements. See "Discovering the @-Keywords" on page 145 for information.

Refer to the IBM IDS *SQL Entry User's Guide* for more information about SQL statements in IDS.

Using the Query Tool to Enter an SQL Statement

You can use the Query tool to create the SQL For Values SQL statement.

To obtain an SQL For Values statement using the Query tool:

1. Create a depiction of the query that generates the desired values from the database.
2. Capture the SQL statement for that query.
3. Copy that SQL statement into the field options window of the Forms Design tool.

Refer to the IBM IDS *Query User's Guide* for more information about using queries. Refer to *SQL Entry User's Guide* for more information about SQL statements in IDS.

Learning About the Default Validation

The Forms Design tool creates some default validation information for fields. When you copy a field from the category window into a form in the main Forms Design window, the Forms Design tool accesses information in the WTDD to determine the default validation information for the selected field.

The Forms Design tool:

Uses WT_COLUMNS to determine the type of data contained in that field

Selects the appropriate choice in the **Accept** field in the field options window

Enters the Minimum and Maximum field values to allow for the range of values

Constraining Information Retrieval

The Retrieval Field Options window constrains information received from the database for the choice and section field types (no matter how the choices were defined for these fields). The **Compare** field determines the method of constraint.

Decide how you want to constrain the information:

Choice Name.

Constrain the information by the choice name given in the Choice Options window. If you choose Choice Name, then the actual title is used as a constraint for information retrieval. For example, in a **Gender** field, the titles Male and Female are used as constraints. See Chapter 8, “Designing Choice Fields” on page 40,” for information about choice names.

Choice Value.

Constrain the information by the choice value given in the Choice Options window. If no choice value has been defined, the relative sequence number of the choice within its field is used, beginning with one. This method is useful when a field contains descriptive names, but you want to use a code to represent the choice. For example, in a **Gender** field you might want to use the value M for Male, and F for Female to constrain information retrieval. See Chapter 8, “Designing Choice Fields” on page 40, for information about choice values.

Database.

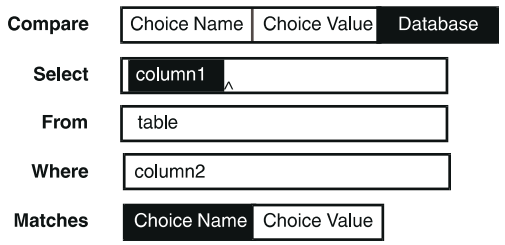
Constrain the information with the database value generated by executing the SQL select statement.

If you select Choice Name or Choice Value, no new options are added. When you select Database, you can specify the dimension table and the column in the database to use as an information constraint. The entries you specify in these fields relate to the method of database storage you chose in the storage area. See Chapter 12, “Storing Data” on page 85, for information about storage options.

To specify database information constraints:

1. Select **Database** in the **Compare** field.

   When you select **Database** in the **Compare** field, new fields appear on the screen that are associated with the database table.



2. In the **Select** field, enter the name of the column in the database table that contains the primary key. This name identifies the column that the Data Entry tool uses to constrain the retrieval.
3. Enter the name of the database table in the **From** field.
4. In the **Where** field, enter the name of the column that contains the choices that appear in the field.
5. Select a choice in the **Matches** field. Choose **Choice Name** to specify that the **Where** field column contains choice names. Select **Choice Value** to specify that the **Where** field column contains choice values.

Chapter 11. Initializing Fields

Enabling you to set the initial value of a field is another way the Forms Design tool gives you flexibility when designing your form. The options described in this chapter appear in the initialization area of the field options windows. The initialization options differ for the three forms, and the differences are described within the chapter. This chapter describes:

- Initialization options for fields
- Initialization options for choice and section fields
- Initialization options for predefined fields
- Field reset options
- Default initialization information

Finding Initialization Options

You can set initialization options for all field types to determine the source and description of the initial value. The initialization options vary depending on the choices that you select in the **Field Type** and **Accept** fields, and the form in which your field resides. The initialization options area for the Entry Field Options window is shown in Figure 16 on page 77.

© Copyright IBM Corp. 1996

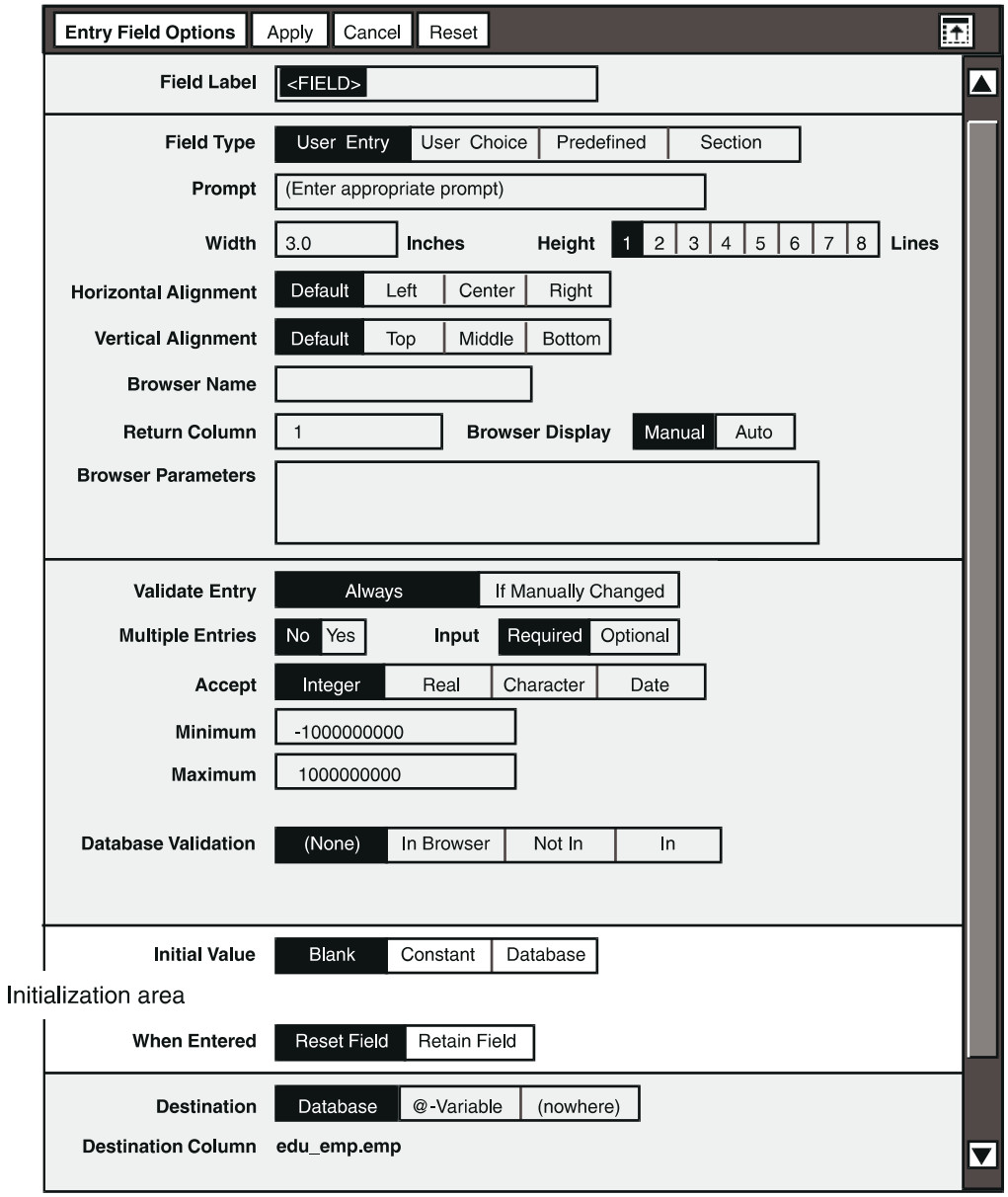


Figure 16. Initialization area in the Entry Field Options window

Initializing Entry and Predefined Fields

The **Initial Value** field for a field and predefined field are essentially the same. However, the choices within the **Initial Value** field vary for a field depending upon the

selection made in the **Accept** field of the validation section. The choices in the **Initial Value** field are different for the following data types:

Integers and real numbers

Character strings

Dates

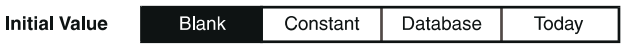
Selecting Initial Values for Entry Fields

The initialization choices for a field vary depending upon the data type of the field. The data type is selected in the **Accept** field. The variations in initialization choices are as follows:

If the data type is an integer or real number, the **Initial Value** field is displayed.

If the data type is a character string, the **Initial Value** field is displayed as shown below.

If the data type is a date, the **Initial Value** field is displayed as shown below.



In all of the previous illustrations, replace the **Blank** choice with **Column** for the update form.

Setting Initial Values for Predefined Fields

For the predefined field, the **Initial Value** field appears.

Replace the **Blank** choice with **Column** for the update form.

Discovering Initialization Methods

The choices explained in Table 13 apply to both entry and predefined fields.

Table 13 (Page 1 of 2). Initial Value field choices for entry and predefined fields

Choice	Initial value
Blank	A blank field appears.
Column	The value from the first row of the column selected to be updated appears. (for update forms only)
Constant	The value from the **Constant Value** field appears.
Database	A database value appears, determined by data from the **Compute** field.
Today	The current date appears.
Now	The current date and time appear.

Table 13 (Page 2 of 2). Initial Value field choices for entry and predefined fields

Choice	Initial value
User	The user's desktop name appears.

When you select **Blank, Today**, **Now**, or **User**, no new options are added. When you select **Column, Constant**, or **Database**, new options are added, and are described in the following sections. The following sections pertain to all three forms except where specified.

Initializing a Field to a Column

You can specify that the value in the first row of the selected column appears in the field when the update form is opened. This option only applies to the update form.

To set a field to a column name:

1. Select **Column** in the **Initial Value** field.
2. Enter a column name into the **Column Name** field.

   For example, to return the first value in the Employee # column of the Personnel table, enter Personnel.employeenumber into the **Column Name** field.

Initializing a Field to a Constant

You can specify a constant value to appear in a field each time a form is opened. Initializing a field to a constant is helpful when you have a field with data that rarely needs to change.

To set a field to a constant value:

1. Select **Constant** in the **Initial Value** field.
2. Enter a value into the **Constant Value** field.

   For example, if your field relates to tax status, enter **Exempt** as the constant value for an **Employee Status** field.

   You can use Forms Design @-keywords in the **Constant Value** field. See the section “Discovering the @-Keywords” on page 145 for information about @-keywords.

Initializing a Field to a Database Value

You can specify that the database provide the initial value for a field. Initializing a field to a database value is helpful when you have a field that relies on variable data.

To set a field to a database value:

1. Select **Database** in the **Initial Value** field.
2. Select a choice in the **Compute** field.

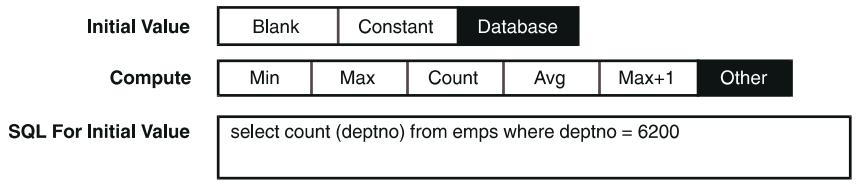
   If you select **Min**, **Max**, **Count**, or **Avg**, the initial field value is the aggregate result of the column name.

If you select **Max +1**, the initial field value is the current maximum value stored in the **Column Name** field plus one. This computation method is useful for primary key fields that contain sequence numbers. For example, to add a new employee into an **Employees** table, you use a predefined field with an initial field value of **Max + 1**.

If you select **Other**, the initial field value is the result of executing the SQL statement specified in the **SQL For Initial Value** field.

3. Enter a value for the **Column Name** field. Enter the name of the column on which the function is performed for the **Min**, **Max**, **Count**, **Avg**, and **Max + 1** choices.

   Select **Other** in the **Compute** field and enter an SQL statement in the **SQL For Initial Value** field. The initialization area appears as shown in the following illustration.



The **SQL For Initial Value** field in the above illustration contains a statement for a field initialized to the headcount in a specific department.

When you write the SQL statement, you must make sure that the statement returns a single value. If the statement does not return a single value, the Data Entry tool displays an error message, and initializes the field to the first value returned by the database server. See Chapter 10, “Validating Data” on page 62, for more information on SQL.

You can use Forms Design @-keywords in the **SQL For Initial Value** field. See the section “Discovering the @-Keywords” on page 145 for information about @-keywords.

Initializing Choice and Section Fields

You can initialize a choice or a section field to one of several possibilities. The options discussed in this section relate to the entry form and update form. The retrieval form does not provide initialization options for choice or section fields.

The initial value in a choice or a section field can be set to:

The first choice in the field for an entry form

The row selected to be updated in an update form

A choice name, choice value, or choice number

A field can also be set to a value from the database. The first value returned from the database upon execution of the SQL statement is used to determine which choice is

highlighted. Initializing a field to a database value is similar to initializing entry and predefined fields to the database.

To initialize a choice or section field:

1. Select **User Choice** or **Section** in the **Field Type** field.

   The **Initial Value** field for a field in an entry form is displayed in the following illustration. (Substitute **Column** for the **Leftmost** choice for an update form.)

2. Select a choice in the **Initial Value** field.

Learning Initialization Methods

Table 14 explains each choice in the **Initial Value** field for choice and section fields.

Table 14. Initial Value field choices for choice and section fields

Choice	Initial value:
Leftmost	The leftmost choice is highlighted.
Column	The first row of the column selected to be updated is highlighted.
Database	The choice that matches the value determined by the SQL For Initial Value statement is highlighted.
Other	The choice name, value, or number specified in the **Initial Choice** field is highlighted.

When you select **Leftmost**, no new options are added. When you select the **Column**, **Database** or **Other** choices, new options are added and are described in the following sections.

Initializing a Field to a Column

This option only pertains to the update form. You can specify that the value in the first row of the column selected to be updated is the initial value.

To set a field to the column:

1. Select **Column** in the **Initial Value** field.

   The initialization area appears.

2. Enter a name in the **Column Name** field.

   For example, to return the first value in the Employee # column of the Personnel table, enter Personnel.employeenumber into the **Column Name** field.

Initializing a Field to the Database

You can specify that the database choose the initial value for a field.

To set a field to a database value:

1. Select **Database** in the **Initial Value** field.

   The initialization area for a field in an entry form appears.

2. Enter an SQL statement in the **SQL For Initial Value** field.

   You must use the name of the choice in the SQL statement rather than the choice value. Also, when you write the SQL statement, you must make sure that the statement returns a single value. If the statement does not return a single value, the Data Entry tool displays an error message, and initializes the field to the first value returned by the database server.

   The value returned by the SQL statement determines which choice is highlighted. If there is no match, the first choice is highlighted. You can also copy an SQL statement from a query. See Chapter 10, “Validating Data” on page 62, for more information on SQL.

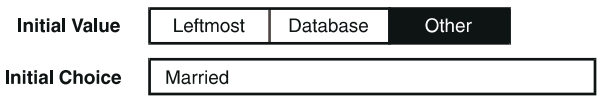
Initializing a Field to a Specific Choice

You can specify exactly which choice you want to appear highlighted when a form is displayed.

To specify an initial choice:

1. Select **Other** in the **Initial Value** field.

   The initialization area is displayed as shown in the following illustration.



2. Enter a value for the **Initial Choice** field. You can enter a choice name, choice value, or choice number.

   When you use this method to initialize a choice or section field, the Data Entry tool determines how to initialize the field when the icon is opened. The Data Entry tool does not know if the value in the **Initial Choice** field that you entered is a choice name, a choice value, or a choice number. Figure 17 on page 83 illustrates how the Data Entry tool initializes a field to the **Initial Choice** value.

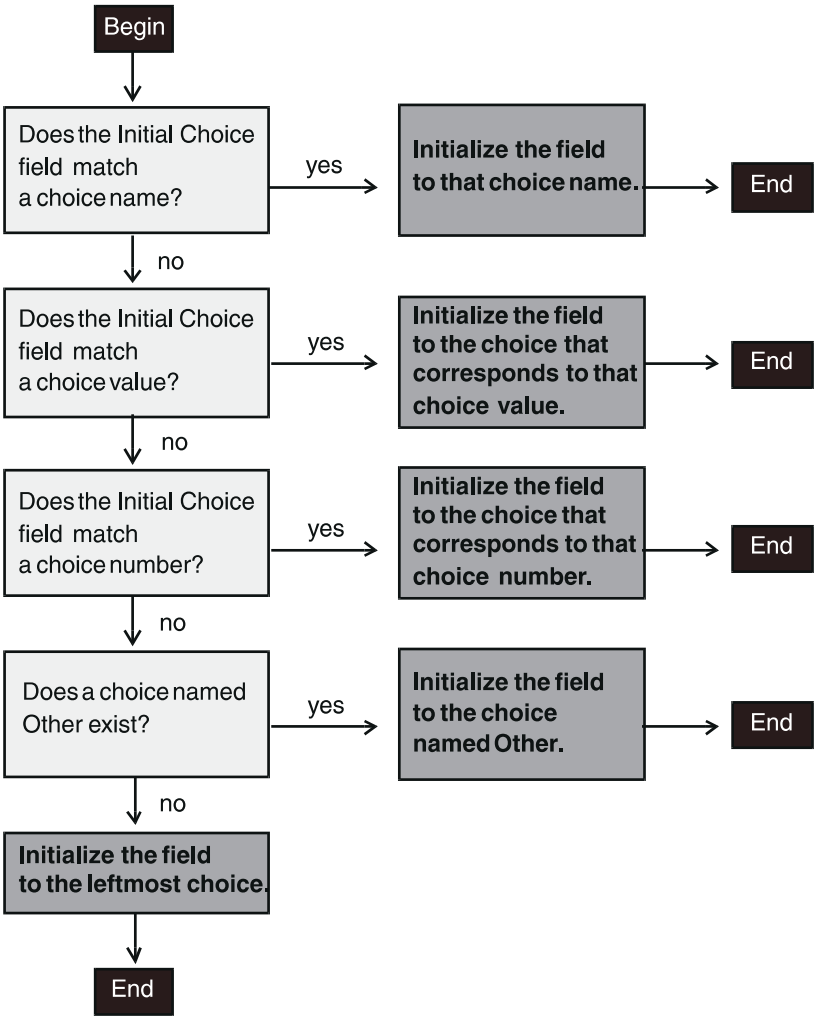


Figure 17. Data Entry tool procedure for initializing a choice or section field

Changing Field Reset Options

You can set each **entry**, **choice**, **predefined**, or **section field** to reset every time the user clicks on the **Enter** button, or you can set the field to retain updated information. This functionality is available only for fields in the entry form.

For example, if the user wants to enter employee information by department, repetition is avoided if the **Department Number** field is set to retain the current department number.

To change field reset options:

1. Select a choice in the **Initial Value** field.

   Regardless of the choice, you can make a selection in the **When Entered** field.

2. Select **Reset Field** in the **When Entered** field if you want the field to return to the initial value after the user enters any new values into the database.

   Select **Retain Field** in the **When Entered** field if you want the field to retain any updated value after the user clicks on the **Enter** button. This choice clears the field only when the user clicks on the **Reset** button in the form window header. This method is useful if the user is entering several records in which one field is the same or similar for all records.

This feature also enables you to place a timestamp on entries. To do so, you create a predefined field whose **Initial Value** is **Now** and whose **When Entered** field is **Reset Field**. Using this method, the time updates when the user clicks on the **Enter** button.

Learning About the Default Initialization

When you copy a field from the category window to a form depiction in the main Forms Design window, the initialization information for that field depends on the form in which the field resides.

The default initialization operates as follows:

If the field is in an Entry Form depiction, the Forms Design tool sets the **Initial Value** field to Blank.

If the field is in an **Update Form** depiction, the Forms Design tool sets the **Initial Value** field to Column. When the update form is used, the Data Entry tool recognizes the appropriate data column.

Except for predefined fields, fields that are copied from a category window into a Retrieval Form depiction do not have initialization information associated with them.

Chapter 12. Storing Data

You can set options to control where the Data Entry tool stores data for the entry and update forms. The options described in this chapter appear in the storage area of the field options windows. This chapter describes:

Storing data in the database

Storing data in *@-variables*

Specifying no data storage

Storing information according to the default

Locating Storage Options

You can specify where you want data stored for the entry and update forms. The **Destination** field in the field options windows gives you three storage choices:

Database

@-Variable

(nowhere)

The dependent fields for these choices vary by the selection made in the **Field Type** field.

To display the field options window:

1. Select a field in a form depiction in the main Forms Design window.
2. Open the field options window. The window looks similar to the Entry Field Options window in Figure 18 on page 86.

© Copyright IBM Corp. 1996

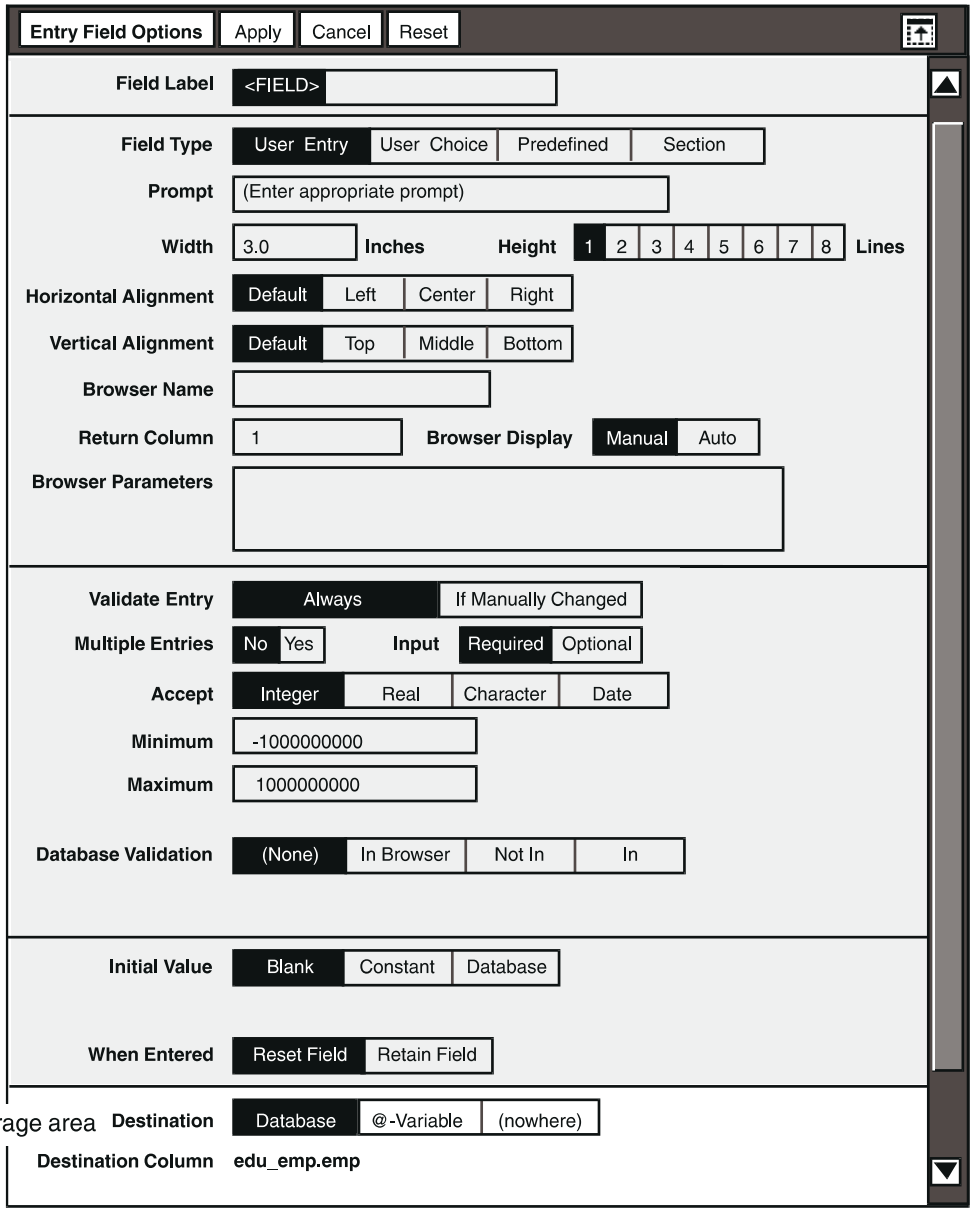


Figure 18. Storage area in the Entry Field Options window

Storing Data in the Database

You can store data in the database for all fields that are copied from the category window. However, the procedure for storing entry and predefined field data is different

from the procedure for storing choice and section field data. The Forms Design tool determines where field data is stored in the database.

Specifying Entry or Predefined Field Storage

To store entry or predefined field data in the database:

1. Select **User Entry** or **Predefined** in the **Field Type** field.
2. Select **Database** in the **Destination** field.

   The **Destination Column** field appears. This field contains the name of the column that stores the data. You cannot change the column name because the Forms Design tool sets this field according to the name of the column that you copied from the category window.

Specifying Choice or Section Field Storage

To store choice or section field data in the database:

1. Select **User Choice** or **Section** in the **Field Type** field.
2. Select **Database** in the **Destination** field.

   The **Destination Column** appears as in the entry and predefined fields, as well as the **Stored Value** field. The **Stored Value** field allows you to choose which value you want stored in the database: choice name, choice value, or database value.
3. Select a choice in the **Stored Value** field. When you select **Choice Name** or **Choice Value**, no new options are added. When you select **Database**, new options are added, and are described in the following section.

   The choices in the **Stored Value** field are explained in Table 15.

Table 15. Database storage options for choice and section fields

Choice	Stored field data
Choice Name	The choice name given in the Choice Options window. For example, in a **Gender** field the choice names might be Male or Female.
Choice Value	The choice value given in the Choice Options window. For example, in a **Gender** field you might want to store the value M for Male, and F for Female. If no choice value has been defined, the relative sequence number of the choice in the field is used, beginning with one. Use the choice value when the field contains descriptive names but you want to store a code to represent the choice.
Database	The database value generated by executing the SQL select statement.

Specifying Database Storage for Choice or Section Fields

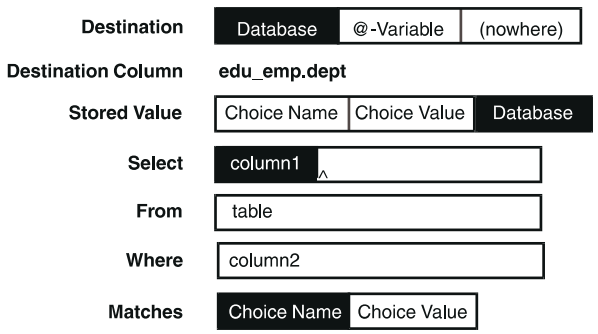
You can specify the dimension table and the column in the database that contain the data and choices to be stored.

To specify database storage information:

1. Select **Database** in the **Destination** field.

2. Select **Database** in the **Stored Value** field.

   When you select **Database** in the **Stored Value** field, new fields appear on the screen that are associated with the database table. The choices for your choice or section field must contain descriptive names generated from the database.



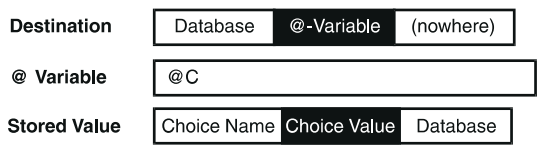
3. In the **Select** field, enter the name of the column in the database table that contains the primary key. This name identifies the data that the Data Entry tool stores in the field.
4. Enter the name of the database table in the **From** field.
5. In the **Where** field, enter the name of the column that contains the choices that appear in the field.
6. Select a choice in the **Matches** field. Choose **Choice Name** to specify that the **Where** field column contains choice names. Select **Choice Value** to specify that the **Where** field column contains choice values.

Storing Data in an @-Variable

When a Data Entry icon is used in a capsule application to set *@-variables*, you can specify that field data be stored in *@-variables*. When you create a Data Entry icon that has fields preset to specific *@-variables*, the Data Entry icon automatically enters current data into those specified User Input Control window.

To store data in an *@-variable*:

Select **@-Variable** in the **Destination** field.



Enter an *@-variable* in the **@-Variable** field.

Select a choice in the **Stored Value** field. See the section "Specifying Choice or Section Field Storage" on page 87 in this chapter for more information about the choices in the **Stored Value** field.

See Chapter 15, "Becoming an Expert Forms Designer" on page 115, for more information on using *@-variables* in a Data Entry icon.

Specifying No Data Storage

You can specify that you do not want to store field data anywhere by selecting **(nowhere)** in the **Destination** field.

Specifying no data storage is useful when:

The Data Entry icon operates inside a capsule application to set *@-variables*, and one or more fields is for display purposes only

You set up hierarchies of fields in such a way that a field is needed only to group subordinate fields

To specify no data storage, select **(nowhere)** in the Entry Field Options window.

Learning About the Default Storage

When you copy a field from the category window into an entry or update form depiction in the main Forms Design window, the Forms Design tool:

Sets the **Destination** field to **Database**

Sets the **Destination Column** field to the name of the column for the field that you copied from the category window

For example, if you copy a field named **Job Code** from the category window into a form in the main Forms Design window, the storage area appears.

The **Destination** field is set to **Database**, and the column name for the **Job Code** field is specified in the **Destination Column** field.

Chapter 13. Working with the Browse Feature

To provide greater flexibility with browse lists, the Forms Design tool enables you to establish a library of browse lists associated with a specific Forms Design icon. You can create and set options for browse lists, use one browse list for several fields, and constrain the information presented in a browse list. This chapter describes:

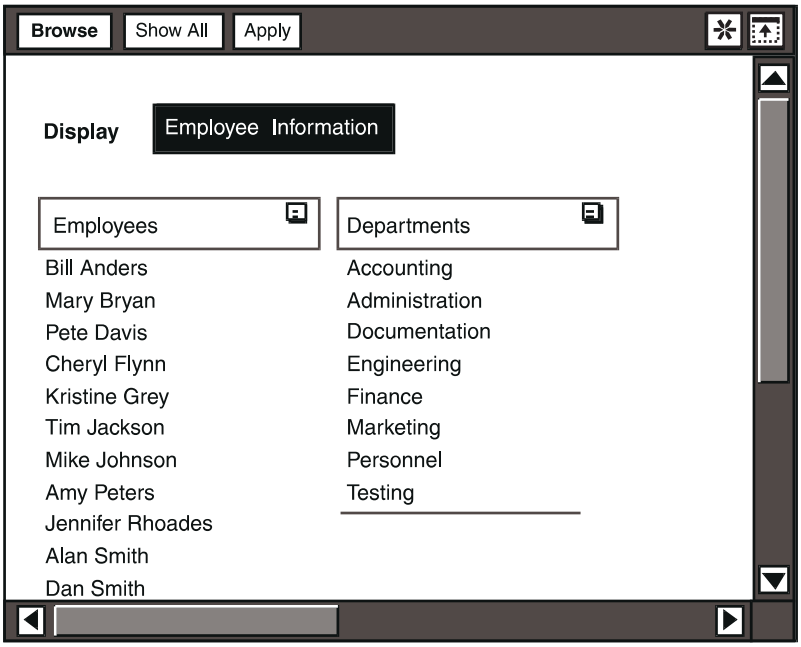
- Creating browse lists
- Updating browse lists
- Adding columns to a custom browse list
- Setting options for columns
- Using one browse list for several fields
- Converting existing Forms Design browse lists

The browse function associated with the Forms Design and Data Entry icons is similar, but not related to the independent Browser icon. Any browse lists you create in the Forms Design and Data Entry icons apply only to those particular icons. For information on the Browser icon, refer to the IBM IDS *Browser User's Guide*.

Setting Up Browse Lists

Browse lists are lists of choices displayed in browse windows. Among other uses, browse lists help users to enter information into fields more accurately. One way the user can display the browse list is by clicking on the **Browse** button in the window header of the Data Entry form. When you have defined a browse list, the browse window appears in the Data Entry form as shown in the following illustration.

© Copyright IBM Corp. 1996



As forms designer, you decide how the browse list is displayed, when the browse list is updated, how the information returned to the browse list is constrained, and many more options. The process for creating and setting options for browse lists has several steps. Each of the following sections describes a part of the process of creating browse lists.

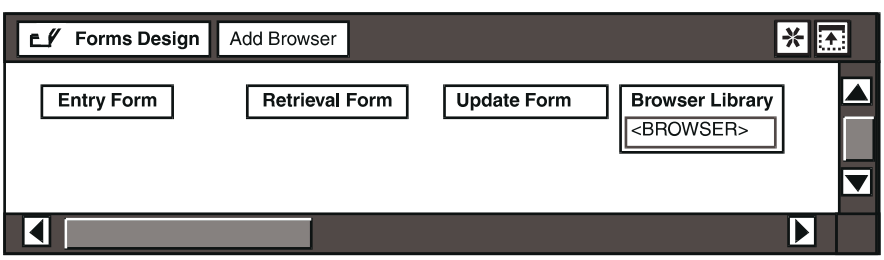
Adding a Browse List to the Browser Library

The main Forms Design window contains a depiction of a browse list library. When you select the Browser Library heading, the buttons in the window header are replaced by a single button, **Add Browser**.

To add a browse list to the Browser Library depiction:

1. Click on the **Add Browser** button.

A field labeled <**BROWSER**> appears in the **Browser Library** depiction.



Setting Preferences for the Browse List

There are several options that you can specify for individual browse lists including the name of the browse list, when to update the browse list, and the location of the SQL statements that define the browse list.

To set the options for the browse list:

1. Select the <**BROWSER>** field unless it is already highlighted.
2. Open the Browser Options window.

   The Browser Options window appears as shown in Figure 19.

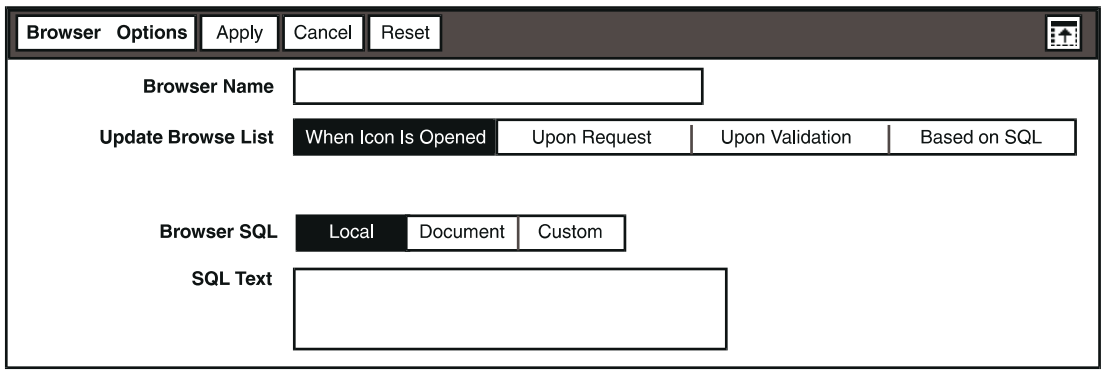


Figure 19. Browser Options window

3. Set the options for the browse list as needed.
4. Close the Browser Options window.

The options within the Browser Options window are described in Table 16, and discussed in the following sections.

Table 16 (Page 1 of 2). Browse list options

Field	Options	Function
Browser Name	N/A	Specify the name of the browse list.
Update Browse List	When Icon Is Opened	Update the browse list every time the icon is opened.
	Upon Request	Update the browse list only when requested by the user.
	Upon Validation	
	Based on SQL	Update the browse list when the browse list is used for validation.
		Update the browse list according to the SQL statement specified in the **SQL For Update Value** field.

Table 16 (Page 2 of 2). Browse list options

Field	Options	Function
Browser SQL	Local	Enter SQL statements that define the contents of the browse list in the Browser Options window.
	Document	Name the document where the SQL statements that define the contents of the browse list are located.
	Custom	Create a custom browse list. See the section "Customizing Browse Lists" on page 96 for more information on custom browse lists.
SQL For Update Value	N/A	Enter SQL statements that determine if a browse list needs to be updated.
SQL Text	N/A	Enter SQL statements that define the contents of the browse list.
Document Name	N/A	Enter the location of the SQL statements that define the contents of the browse list.
General Qualifier	N/A	Specify a constraint on the browse list. The entry in this field is equivalent to a "where" clause in SQL.

Naming the Browse List

The name you enter for the browse list should indicate the contents. When you look at a collection of browse lists in the Browser Library depiction, you need to be able to discern easily the contents of each browse list. For example, if a browse list contains an employee's name, address, and phone number, then name the browse list "Employee Contact Information." Enter this name into the **Browser Name** field.

To associate the browse list with a field, you enter the browse list name in the **Browser Name** field of the field options window. See Chapter 7, "Constructing Entry Fields" on page 33, for more information about browse list options in the field options windows.

Updating the Browse List

Select a choice in the **Update Browse List** field to control when a browse list is updated.

Depending upon how often the database changes, you can specify that the browse list be updated:

When Icon Is Opened

The browse list is updated every time the Data Entry icon is opened and the **Browse** button is clicked on for the first time.

Upon Request

The browse list is updated when **Update Browse Lists** is selected from the Data Entry icon Special menu.

Upon Validation

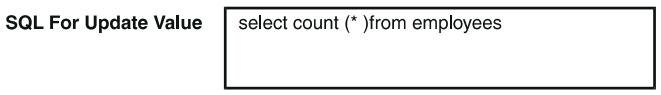
The browse list is updated when it is used for validation.

Based on SQL

The browse list is updated according to the SQL statements specified in the **SQL for Update Value** field.

When you select Based on SQL, an **SQL for Update Value** field appears. You enter an SQL statement into this field that gives the Data Entry tool a way to automatically determine when a browse list needs to be updated.

For example, to update the browse list every time the first value has changed, enter an SQL statement as shown in the following illustration.



When the database is queried and returns a value, that value is compared against the value that was returned the last time the tool was opened and the SQL statement was issued. If the values are different, the Data Entry tool clears all the values in the browse list. The browse list is then updated when the user clicks on the **Browse** button. You should use any measure appropriate to your browse lists when creating this SQL statement.

If the database changes frequently, then the browse list should be updated frequently. However, updating the browse list is time consuming, so only update as necessary. See the section “Determining Data Entry SQL Performance” on page 141 for more information on performance issues related to browse lists.

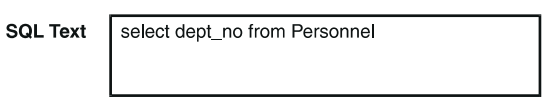
Determining the Browser SQL

When deciding on the SQL with which to define browse lists, you should consider consolidating the fields that can share the same “where” clause. See the section “Using One Browse List for Several Fields” on page 102 for more information. Select a choice in the **Browser SQL** field to control how the relevant information is passed from the database into the browse list.

There are three choices in the **Browser SQL** field:

Local

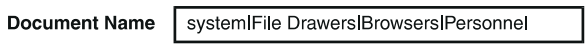
You specify SQL statements in the **SQL Text** field that provide the table name and the names of specific columns to access for the browse list. For example, if you want the browse list to contain the Department # column from the Personnel database table, enter a statement into the **SQL Text** field as shown in the following illustration.



Document

You specify the name of the document in the **Document Name** field that contains the SQL statement. By storing the SQL in a separate document, you can use the SQL statement for many different Forms Design icons. This method also helps with maintenance of the browse lists in case of changes.

For example, if the name of the document is "Personnel" in the Browsers File Drawer, enter the document name and path as shown in the following illustration.

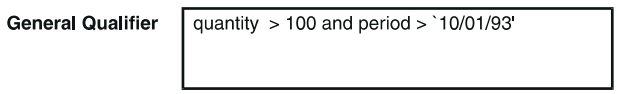


You can store the document on the desktop, in a container such as a Folder icon or File Drawer, in your capsule application, or any other location within IDS. Just be sure to provide the correct path to the document. IDS will search for the document in the current container or capsule icon first, then the desktop, and finally in the System|File Drawers.

Custom

You create your own columns for the browse list. This method is especially useful when you want to specify a formula which alters the information that appears in the browse list.

When you specify Custom, you can enter a “where” clause in the **General Qualifier** field that constrains the data allowed in the browse list (you performed the “select” by adding columns to the browse list). For example, the following illustration shows a browse list constrained to only return data for October 1, 1993 with quantities greater than 100.



You can set up the browse list so that it can be used by different fields to access different parts of the browse list. The @-keywords @PARM1, @PARM2, ...@PARM10 allow you to set up a constraint without specifying the conditions in the **General Qualifier** field. The specific conditions for each field are then given in the **Browser Parameters** field of the field options window. See the section “Setting Up Conditional Expressions with @PARMs” on page 126 for more information.

If you select either Local or Document, the browse list contains the columns specified by the SQL statements. If you select Custom, then you decide to create your own columns for the browse list. See the section “Customizing Browse Lists” on page 96 for information about creating your own columns.

Customizing Browse Lists

Customizing browse list columns is useful for several reasons:

You can combine columns from different tables in the same browse list. By copying columns directly from the category window to the 'Browser' columns heading, you can combine columns from different tables without becoming confused by complicated SQL statements.

You can tailor the data type of the column according to the WTDD definition. This feature helps you show data in the most appropriate form. For example, you can show real numbers as integers.

You can perform calculations on the data retrieved for the browse list, and display the altered information.

Choosing a Method of Customizing

You can customize a browse list two ways:

The category window

Copy columns from the category window to the 'Browser' columns heading. Use this method when you want specific columns from database tables, but you do not want to write SQL statements. This method does not allow you to alter the values displayed by the browse list, or determine the display type for the information. The Browser Column Options window specifies the table and column heading for you as well.

The Create Column button

Add columns to the 'Browser' columns heading by clicking on the **Create Column** button. Use this method when you want to create a column that is not defined in the WTDD. Also use this method if you want to specify a formula which alters the information you retrieve from the database for the browse list. For example, if the table you are accessing contains prices for items without sales tax figured in, you can display the price with the sales tax in the browse list.

For both of these methods, you must specify Browser Options window. The **Custom** choice informs the Forms Design tool that you are creating your own columns using one of the methods outlined in this section. If you specify **Local** or **Document** in the **Browser SQL** field of the Browser Options window, the 'Browser' columns heading does not appear.

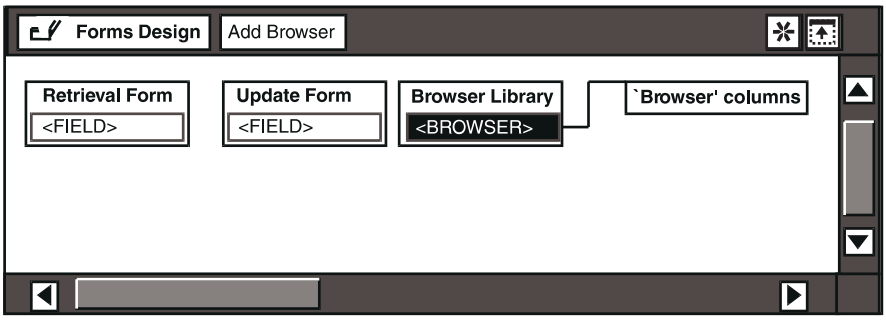
Adding Columns to the Browse List

As shown in the previous section, there are two ways to add columns to the browse list. Either way you choose to add columns, you follow the same steps to reach the 'Browser' columns heading. The following example shows how to add columns using the **Create Column** button.

To add columns to the browse list:

1. Be sure to set the **Browser SQL** field in the Browser Options window to **Custom** so that you will be able to add columns to the browse list.
2. Double-click on the browse list field in the Browser Library depiction.

   The 'Browser' columns heading appears as shown in the following illustration.



The 'Browser' columns heading takes on the name of your browse list.

1. Select the 'Browser' columns heading.

   The **Add Browser** button in the window header is replaced by the **Create Column** button.
2. Click on the **Create Column** button.

   A field labeled <**COLUMN>** appears under the 'Browser' columns heading.

Designing Browse Columns

When you have added a column to the browse list, you can set several options that affect the display, sorting method, and contents of the column.

To set options for the browse column:

1. Select the column under the 'Browser' columns heading.
2. Open the Browser Column Options window.

   The Browser Column Options window, for a column created with the **Create Column** button, appears as shown in Figure 20 on page 98.

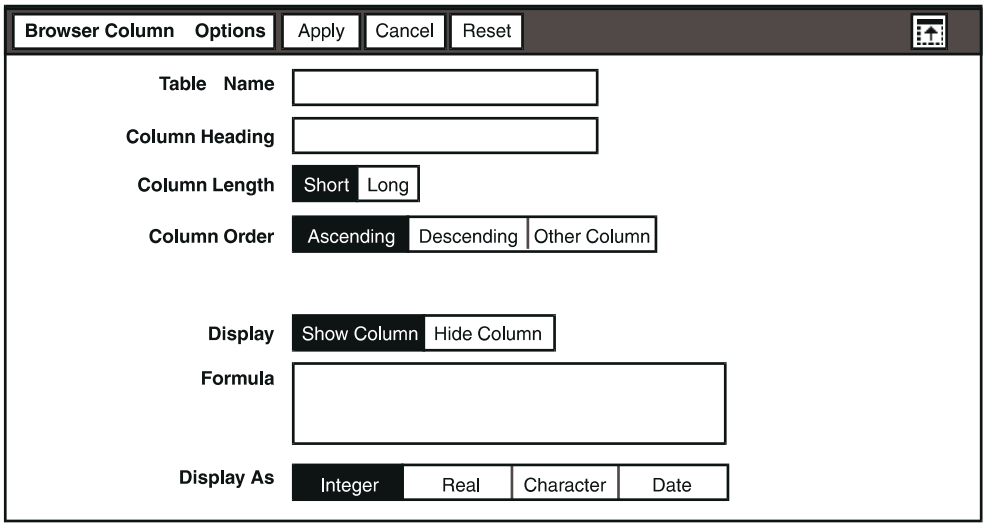


Figure 20. Browser Column Options window for a column created with the Create Column button

3. Set the options as needed.
4. Close the Browser Column Options window.

If you copy the column from the category window, the Browser Column Options window looks slightly different.

Notice that the information for the **Table Name** and **Column Heading** fields are supplied by the Forms Design tool. Note also that the **Formula** and **Display As** fields are missing from this window. When you copy fields from the category window, you cannot alter the information retrieved for the browse list or decide what data type the browse list uses to display the information. However, you can change the column name given in the **Column Heading** field.

The various options available in the Browser Column Options window are described in Table 17 and discussed in the following sections.

Table 17 (Page 1 of 2). Browse list column options

Field	Options	Function
Table Name	N/A	Specify the name of the database table to be accessed by the browse list column. The information for this field is supplied by the Forms Design tool if you copy the column from the category window.
Column Heading	N/A	Specify the display name for the browse list column.
Column Length	Short	Estimate that the length of the column does not exceed the 100 item display limit for the browse list window.
	Long	Estimate that the length of the column is likely to exceed the 100 item display limit for the browse list window.

Table 17 (Page 2 of 2). Browse list column options

Field	Options	Function
Column Order	Ascending	Sort the column into ascending order.
	Descending	Sort the column into descending order.
	Other Column	Sort the column according to the column you specify in the **Order by Column** field.
Order by Column	N/A	Specify the name of the column to be used as a basis when sorting the current column.
Display	Show Column	Display the column in the browse list.
	Hide Column	Conceal the column in the browse list.
Formula	N/A	Enter SQL statements as a formula to generate customized columns to retrieve information into the browse list.
Display As	Integer	Display numeric data as whole numbers.
	Real	Display numeric data as numbers with decimal points for which you can specify a precision in the **Precision** field.
	Character	Display data as strings of alphanumeric characters.
	Date	Display data as dates for which you can specify a precision in the **Resolution** field.
Precision	0000.1 - .00005	Determine the placement of the decimal point for data that is defined as a real number.
Resolution	Day	Display the date in days.
	Week	Display the date in weeks.
	Quad Week	Display the date in four week periods.
	Month	Display the date in months.
	Odd BiMonth	Display the date as 1st and 3rd month periods.
	Even BiMonth	Display the date as 2nd and 4th month periods.
	Quarter	Display the date in quarter years.
	Year	Display the date in years.

Predicting Column Length

The browse list has a limit of 100 items that it can store in workstation memory and display in a single column. If a column exceeds this 100 item limit, the browse list does not display any items for the column until enough constraints have been placed on the data so that the number of items returned in the column does not exceed the capacity of workstation memory.

As forms designer, you must predict the number of items that will be returned in the columns that you create, and decide whether the column exceeds display limits. In the **Column Length** field, you choose the selection that is most appropriate.

Select **Short** if you think that the column will return less than 100 items, and IDS will store and display the column's contents from memory. Select **Long** if you think the column will return more than 100 items. For long columns, IDS does not store the column's contents in memory. Rather, IDS will retrieve and display the column's contents when the column has been constrained by choices in other columns, and the number of items to be retrieved in the column has been constrained to below 100.

Always begin with the **Short** choice in the **Column Length** field. If you later discover that some columns are using too much memory, or if you experience performance problems, you can specify that the columns with the most items are **Long**. Also remember that the width of a column can affect the amount of memory used. For example, a column that is 250 characters wide requires a lot of memory, and slows down performance both while defining the browse list and during later use.

Ordering Columns

You determine how each column in the browse list is sorted by selecting a choice in the **Column Order** field.

You can sort the column by three ordering methods:

Ascending

The column appears in ascending order – for example, Apple, Banana, Cherry; and 1,2,3.

Descending

The column appears in descending order – for example, Cherry, Banana, Apple; and 3,2,1.

Other Column

The column appears in the order of another column. This method keeps items that are related in the same order, but not necessarily in the same row. If an item appears in two rows of the column, it is only listed once in the proper order. For example, assume that the employees and departments actually correlate as shown in Table 18.

Table 18. Actual correlation of employees and departments

Employee	Department
Jan Adams	Marketing
Dave Jones	Personnel
Sharon Korth	Sales
Ben Marks	Marketing
Mary Smith	Engineering

To correlate the department to the employee, you define column two with the **Column Order** field set to **Other Column**, and specify column one in the **Order by Column** field. The table appears as shown in Table 19 on page 101.

Table 19. Column two sorted by column one

Employee	Department
Mary Smith	Marketing
Dave Jones	Personnel
Sharon Korth	Sales
Jan Adams	Engineering
Ben Marks	

Notice that there are no duplicate instances of a department; but the departments now appear in the order of the employee names. The only limitation on the use of the column order feature is that the column listed in the **Other Column** field must be in the same browse list as the column you are defining.

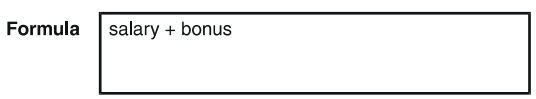
Determining Column Display

You determine whether or not to show a column in the browse list by selecting a choice in the **Display** field.

This feature is useful when you need to sort a column according to another column which you wish to hide. For example, if you want to sort employee names according to salary groups, but do not want to show the salaries.

Entering a Formula

By using the **Formula** field, you can perform calculations on the data retrieved by the browse list, and display the updated information. For example, if you have a browse list column that returns the salary of each employee, and you want to display the salary plus bonus, you enter a calculation into the **Formula** field. The calculation that returns the sum of salary plus bonus is shown in the **Formula** field in the following illustration.



The **Formula** field is useful when you want to place different constraints on individual browse list columns. This option is available only when you create columns with the **Create Column** button.

Setting Data Type

When you customize your columns with the **Create Column** button, you determine the data type that the browse list uses to display information.

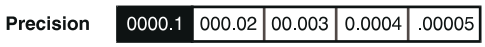
The choices in the **Display As** field are:

Integer

Displays numeric data as a number with no decimals – for example, 1, 20, or 323.

Real

Displays numeric data as a number with a decimal point – for example, 5.21. When you select this choice, the **Precision** field appears as shown in the following illustration.



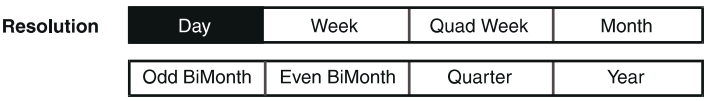
Select a choice based on the number of decimal points you want.

Character

Displays data as a string of alphanumeric characters.

Date

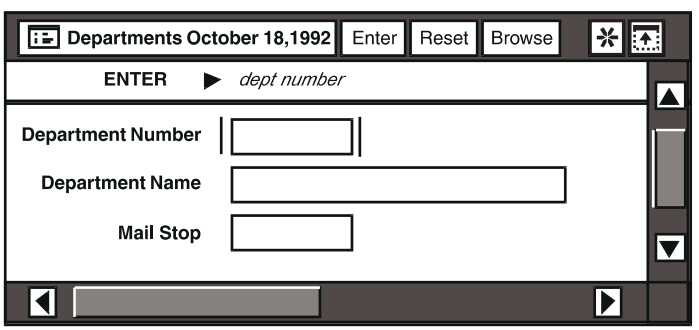
Displays data as a date. When you select this choice, the **Resolution** field appears as shown in the following illustration.



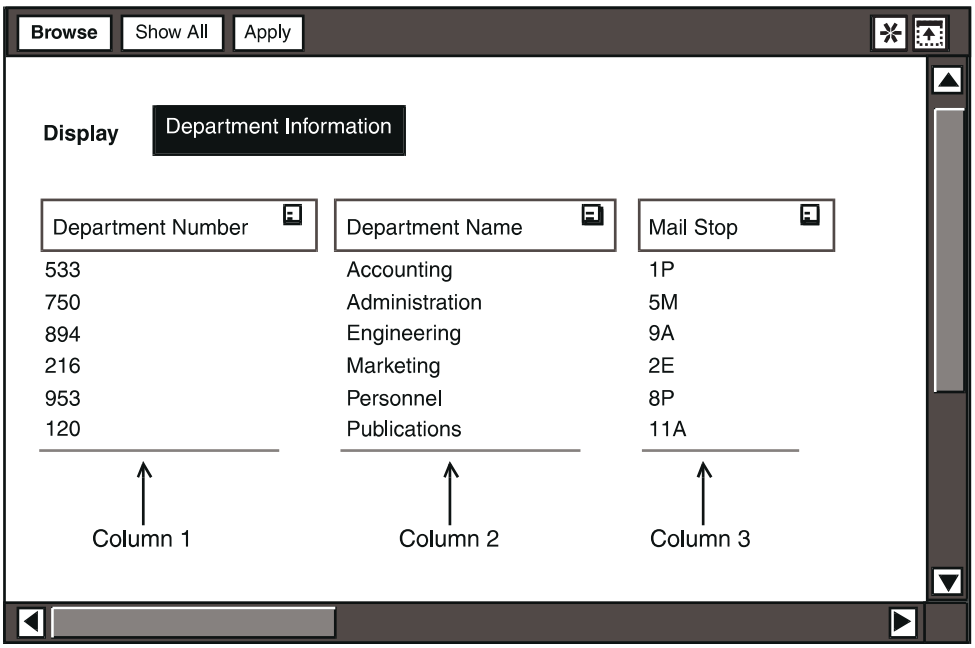
Choose the precision with which you want the browse list to display the date. See the section “Creating Date Value Ranges” on page 68 for more information about the choices offered by the **Resolution** field.

Using One Browse List for Several Fields

With the enhanced browse list functionality, you can use one browse list for several fields. For example, suppose you have an entry form that looks similar to the following illustration.

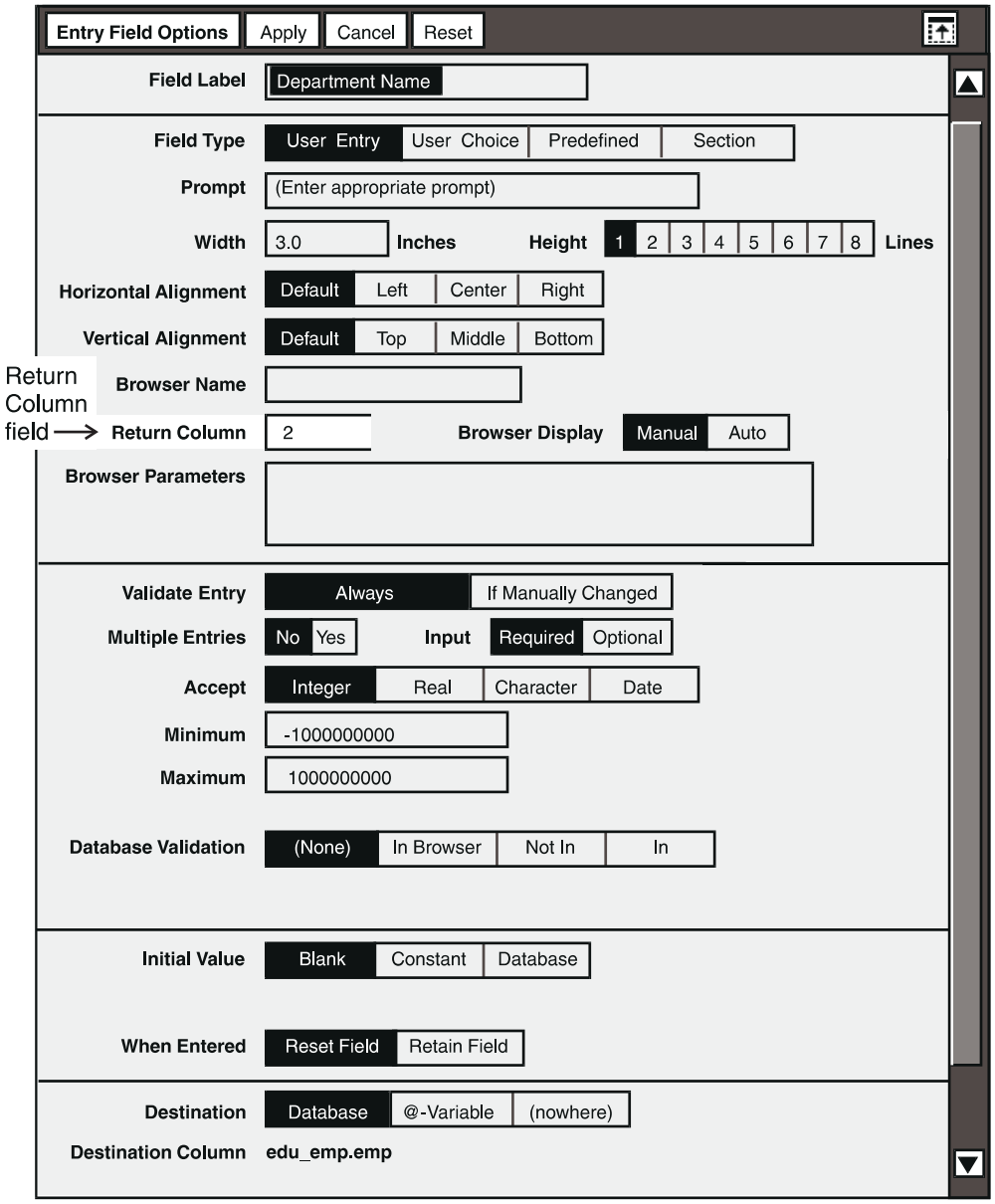


You can set up one browse list to accommodate each field within this form. Simply create a browse list with columns for the department number, department name, and mail stop as shown in the following illustration.



As shown in the above illustration, Department Number is column 1, Department Name is column 2, and Mail Stop is column 3. To use this browse list for all three fields, you must specify the correct column to return information to the specific fields. Enter the column number in the **Return Column** field in the field options window.

For example, to specify that the item in the Department Name column is returned to the field, you enter **2** in the **Return Column** field as shown in the following illustration.



See Chapter 7, “Constructing Entry Fields” on page 33, for more information on browse list options in the field options window.

Converting Existing Forms Design Browse Lists

When a Forms Design browse list from a previous DIS release is encountered, a new browse list is created that complies with Release 1. The name of the browse list remains the same, but the **Return Column** field in the field options window is set to 1, and the **Browser SQL** field is set to Local.

Returning Information from the Right Column

You can access all columns in the browse list by specifying the column number in the **Return Column** field. Thus, when converting an existing Forms Design browse list, the **Return Column** field needs to be set to 1 because the existing browse lists are already set up with the valid information in column one.

Understanding the Browser SQL Choice

The **Browser SQL** field is set to Local because the SQL text is taken from the old Forms Design browse list and placed in the **SQL Text** field. This method sends the SQL statements as they are, and the Data Entry tool is not affected.

Chapter 14. Enhancing Forms with Design Techniques

Placing fields side-by-side, separating areas of a form, and adding special instructions are several design techniques Forms Design offers you. Use these techniques to design effective and efficient forms. This chapter describes:

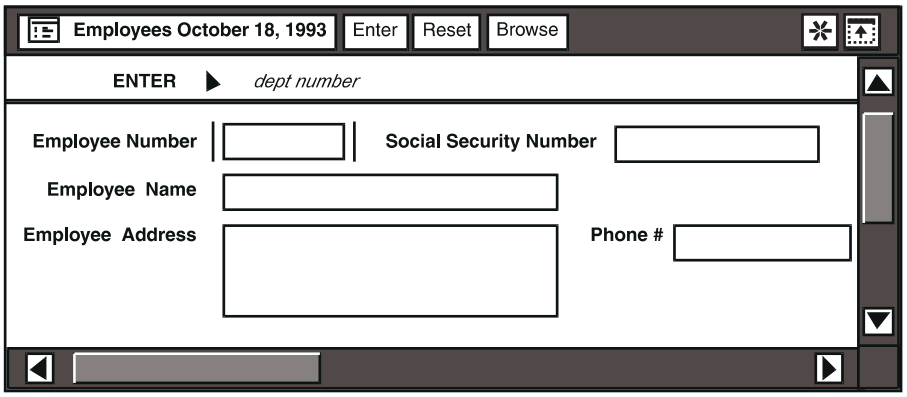
Creating groups to place fields side-by-side

Using lines to separate fields in a form

Embedding special instructions in a form

Grouping Fields

The group feature gives you greater flexibility when creating forms. Grouping fields together saves form space by enabling you to place fields, or groups of fields, side-by-side.



In the previous illustration, the Data Entry form saves space by placing the **Employee Number** and **Social Security Number** fields next to each other. These two fields are part of a group.

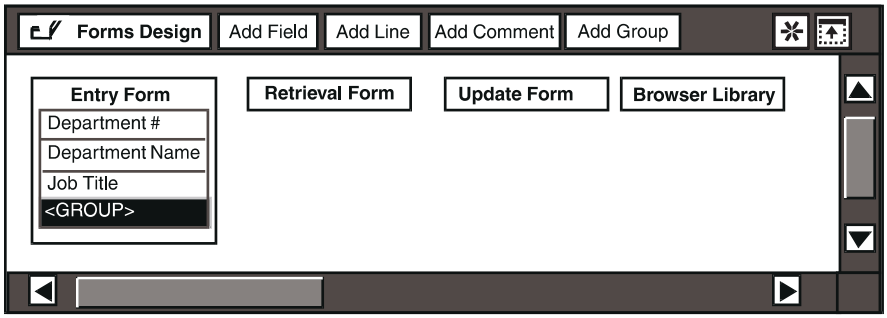
Creating Groups

To create a group:

1. Click on the **Add Group** button in the main Forms Design window header.

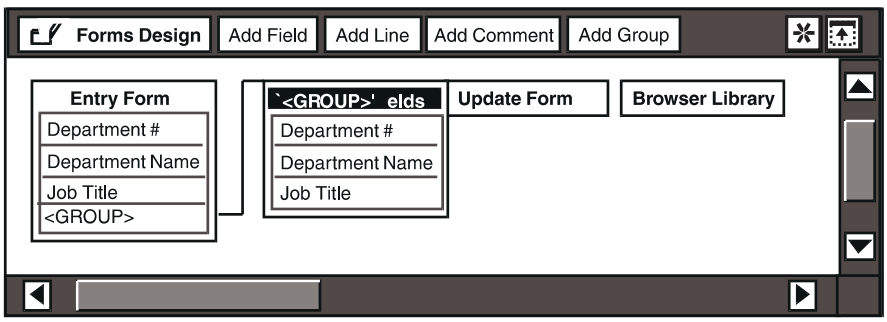
   A field labeled <**GROUP>** is added to the last form you selected.

© Copyright IBM Corp. 1996



2. Select the highlighted <**GROUP>** field. A heading, '<GROUP>' fields, appears.
3. Select the <**GROUP>' fields** heading. Add fields to the group by copying fields to the <GROUP>' fields heading, or by clicking on the **Add Field** button in the main Forms Design window header.

   For example, copy the **Department #**, **Department Name**, and **Job Title** fields to the '<GROUP>' fields heading.



4. Double-click on the <**GROUP>** field to close it.

Designing Groups

Now that you have added fields to the group, you can set options for the group that determine how the fields are placed on the form.

To set options for a group:

1. Unless it is already highlighted, select the group field for which you want to set options.
2. Open the Group Options window.

   The Group Options window appears as shown in Figure 21 on page 108.

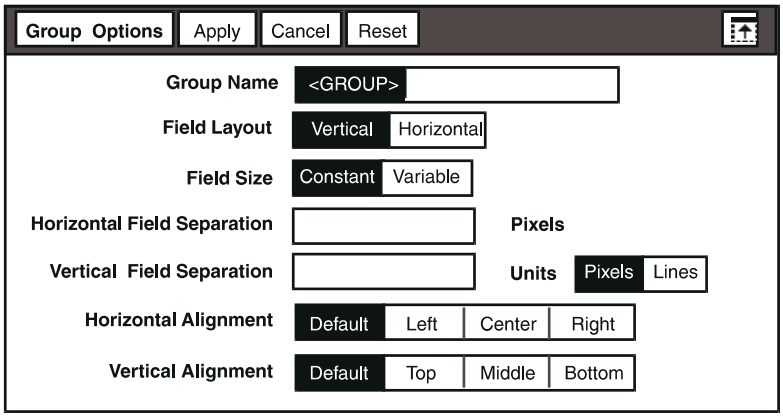


Figure 21. Group Options window

3. Change the options as needed.
4. Close the Group Options window.

With the group feature, you can create groups within groups. By using this technique, you can create several sub-groups within one group. Experiment with the functionality of groups to enhance your forms.

Table 20 explains the options available within the Group Options window.

Table 20 (Page 1 of 2). Group field options

Field	Options	Function
Group Name	N/A	Enter the name of your group.
Field Layout	Vertical	Stack the fields within the group.
	Horizontal	Place the fields side-by-side within the group.
Field Size	Constant	Make each field space as big as the space occupied by the longest and widest fields.
	Variable	Make each field space differ according to the space needed by each field.
Horizontal Field Separation	N/A	Determine the relative amount of space between fields that are placed side-by-side.
Vertical Field Separation	N/A	Determine the relative amount of space between fields that are stacked.
Units	Pixels	Measure the vertical field separation in pixels.
	Lines	Measure the vertical field separation in lines.

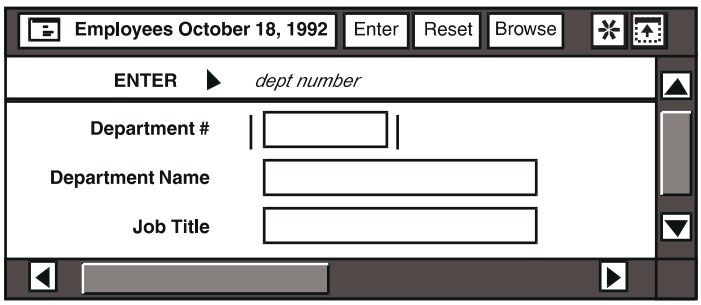
Table 20 (Page 2 of 2). Group field options

Field	Options	Function
Horizontal Alignment	Default	Align the group according to the system default.
	Left	Align the group flush left in its space.
	Center	Center the group horizontally in its space.
	Right	Align the group flush right in its space.
Vertical Alignment	Default	Align the group according to the system default.
	Top	Align the group at the top of its space.
	Middle	Center the group vertically in its space.
	Bottom	Align the group at the bottom of its space.

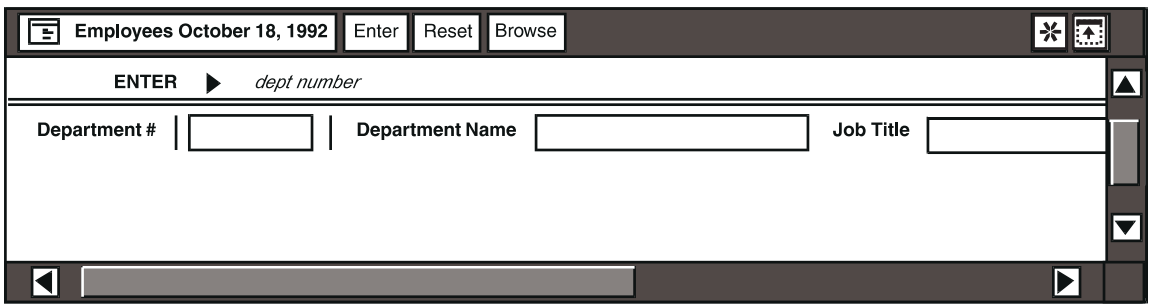
Placing Fields

You decide whether to stack the fields or place them side-by-side within a group.

If you define the **Field Layout** as **Vertical**, the fields appear stacked in the entry form as shown in the following illustration.



If you select the **Horizontal** choice, the fields appear side-by-side. The following illustration shows fields placed horizontally within their group.



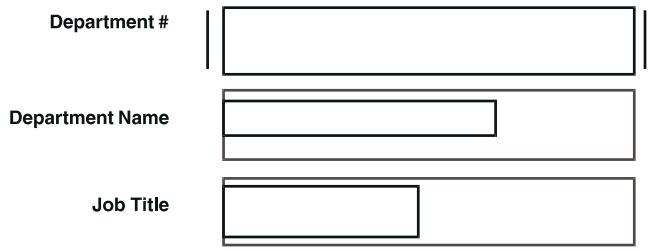
The **Horizontal** choice is especially useful if you have a form with many fields that have small lengths. Instead of scrolling down a very long form, you can condense information by grouping fields and specifying horizontal layout.

For an example of how you can design a form by placing fields horizontally or vertically, see Chapter 16, “Designing Complex Forms” on page 129.

Setting the Field Size

You determine the space occupied by the fields in your group. You can allocate space for fields according to the space occupied by the tallest and widest fields, or you can allocate space based on the needs of individual fields.

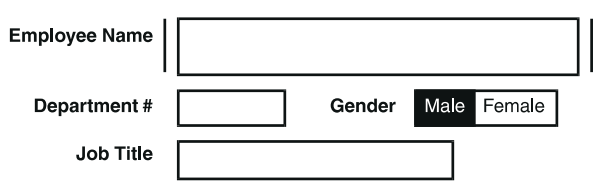
To allocate space based on the space occupied by the tallest and widest fields, select **Constant** in the **Field Size** field. The space taken by every field is the same, and is equal to the space occupied by the tallest and widest fields in the group. The next example shows a group in a form with a constant field size.



In the above illustration, the **Department #** field is the tallest and the widest field; therefore its size determines the space the rest of the fields in the group can occupy. The dotted lines represent the space the **Department Name** and **Job Title** fields could occupy.

If you want to align fields horizontally or vertically within a group, you must specify **Constant** in the **Field Size** field. See the section “Aligning Horizontally and Vertically” on page 35 for more information.

To vary the space allocated to each field, select **Variable** in the **Field Size** field. When you have many fields, selecting **Variable** helps save space; especially if you place the fields side-by-side. The space for each field is determined by the width and height required for the particular field. The next illustration shows a group with a variable field size.



Setting Field Separation

You determine how much space separates fields within a group.

The **Horizontal Field Separation** field specifies the amount of space, in pixels, between fields that are placed side-by-side. The **Vertical Field Separation** field specifies the amount of space between fields that are stacked. You set the unit of measurement for vertical field separation in the **Units** field.

Aligning Horizontally and Vertically

You determine the horizontal and vertical alignment of a group by selecting choices in the **Horizontal Alignment** field and **Vertical Alignment** field. See the section “Aligning Horizontally and Vertically” on page 35 for more information.

Using Lines to Separate Fields

When you create a form with many fields, you can give the form structure with lines. For example, the form in Figure 22 has a dotted line to separate the first four fields from the last two fields.

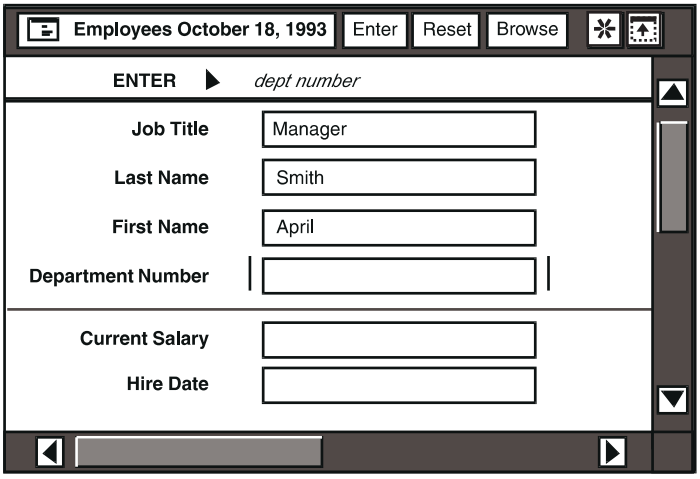


Figure 22. Dotted line in a form

Creating Lines

To separate fields using a line, click on the **Add Line** button in the window header. A **----LINE----** field appears at the bottom of the form.

If the line is in the wrong position, you can move it below the field that you want it to follow.

Changing the Line Type

To change a line type:

1. Select the **----LINE----** field.
2. Open the Line Options window shown in Figure 23.

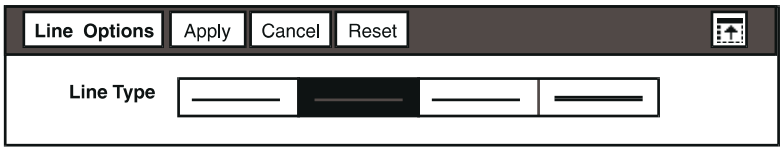


Figure 23. Line Options window

3. Select a choice from the **Line Type** field.

   By default, a line that you add with **Add Line** is the second choice in the **Line Type** field shown in Figure 23. This choice is displayed as a faint solid line.

4. Close the Line Options window.

   If you want to add a blank space following the line you just added, use a comment field. You can create a comment field with the **Add Comment** button.

Embedding Special Instructions

There might be times when you want to embed special instructions in a form. For example, if you have a form that sets *@-variables*, you can include instructions to tell the user how to run the capsule application.

Figure 24 on page 113 shows an entry form that contains special instructions.

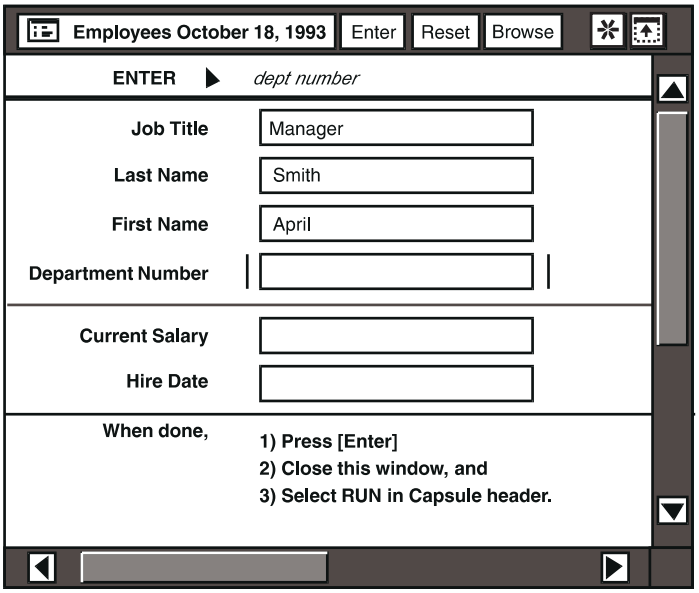


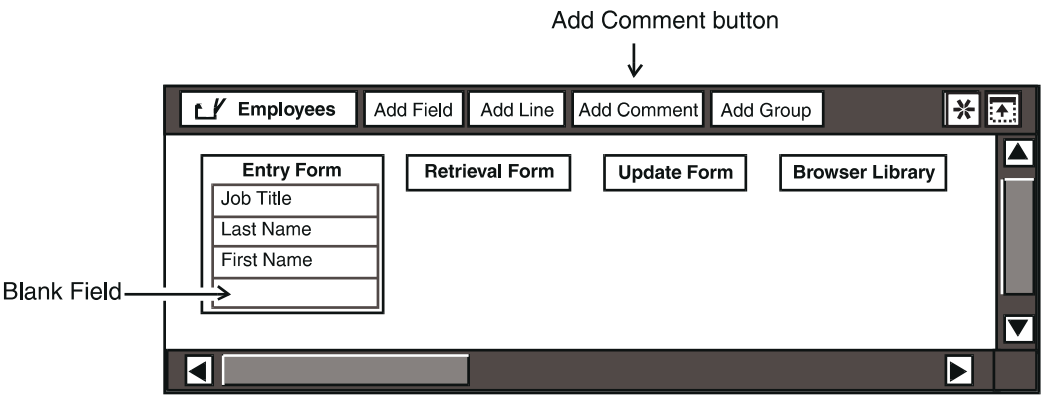
Figure 24. Special instructions embedded in a form

Each step in the instructions corresponds to a field in the Entry Form depiction in the main Forms Design window.

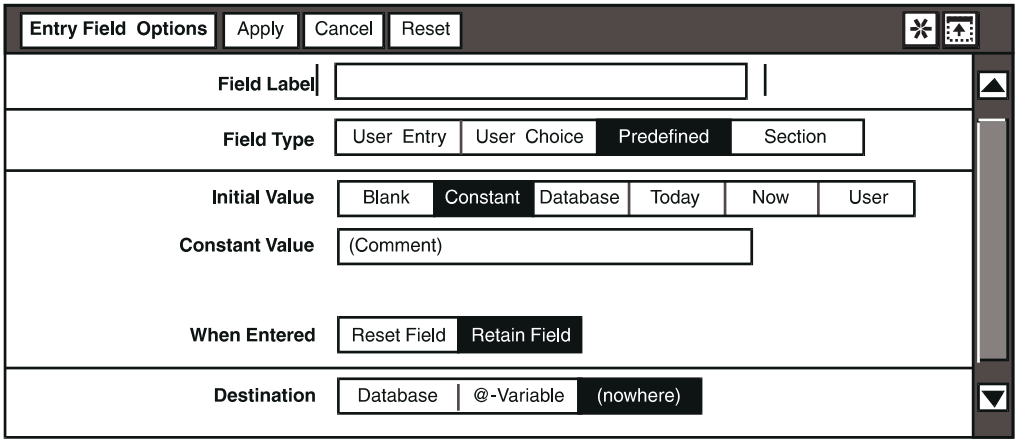
To add comments to a form:

1. Click on the **Add Comment** button.

   A blank field appears at the bottom of the form depiction.



2. Select the blank comment field, if it is not already highlighted.
3. Open the field options window.



4. Enter a label in the **Field Label** field if you want your instructions to align with the other fields in the form.

   For example, to create the instructions shown at the bottom of Figure 24 on page 113, When done, was entered into the **Field Label** field.

5. Enter an instruction into the **Constant Value** field.

   In this example, 1) Press Enter was entered.

6. Select **Retain Field** in the **When Entered** field.
7. Select **(nowhere)** in the **Destination** field.
8. Close the field options window when complete.
9. Repeat this procedure to add any additional comments or instructions.

   In this case, the **Field Label** field was left blank for the additional instructions so that all instructions are left aligned.

Chapter 15. Becoming an Expert Forms Designer

As you become experienced with the Forms Design tool, you might consider using its more technical features to set *@-variables* within a capsule application, or use @-keywords to determine the name of your Data Entry icon. This chapter describes the following:

Using primary key fields in forms

Displaying database information for fields

Creating a form to set *@-variables*

Generating post-entry SQL

Specifying Forms Design @-keywords

Enhancing fields in retrieval forms

Modifying Forms Design tools with scripting

Using column and table name indirection

General Form Considerations

The following sections describe general information that you should consider when you initially design a form.

Using Primary Key Fields in Forms

Forms that allow a user to insert, delete, or update rows in a database table must be carefully designed to specify the primary key for the table. Otherwise, the form might not include the appropriate constraints, and the user could alter more rows than intended. The primary key fields need not be in the form itself; the values can be specified in post-entry SQL. See the section “Generating Post-Entry SQL” on page 120 in this chapter for more information.

See the section “Generating the Choices Dynamically” on page 50 for information on primary keys.

Limiting the Number of Tables Each Form Represents

Although you can design an entry form with columns from more than one table, do so with caution. Each entry form should refer to columns from only one table unless you are inserting the data with post entry SQL.

Retrieval and update forms only access one table. If you create a retrieval form or update form with columns from more than one table, the user receives an error message when trying to use the form.

If a data entry task involves multiple tables, you can do one of the following:

Create one Data Entry icon for each table, and place all the Data Entry icons in a folder for distribution.

© Copyright IBM Corp. 1996

Use post-entry SQL to update multiple tables. See the section “Generating Post-Entry SQL” on page 120 for more information.

Using Views in Forms

Be very cautious when you design forms based on views. The view might contain columns from more than one table, or it might not be possible to use the view to insert or update data. Refer to the IBM IDS *Administering Databases with IDS* publication for information on views.

Displaying Database Information for Fields

You can display database information specific to each entry or predefined field, or for multiple fields within a form.

Displaying Information for Single Fields

To display database information for a single field (entry or predefined):

1. Double-click on an entry or predefined field in the Forms Design window.

   An information box appears as shown in Figure 25.

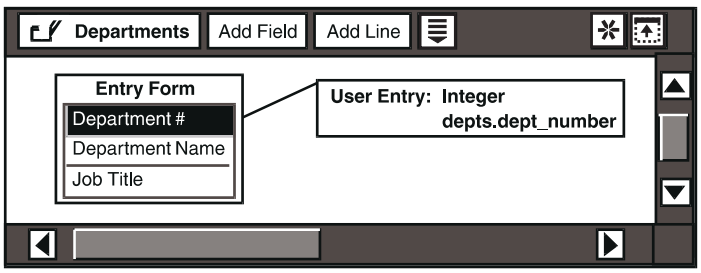


Figure 25. Information box in the Forms Design window

   The information box contains the field type, the acceptable data type, and the name of the database column to which the selected field corresponds. In this example, Department # is defined as a **User Entry** field, with an acceptable data type of Integer, and database column depts.dept_number.

   You can obtain the same information by double-clicking on the desired column in the table shown in the category window.

2. Select the field again.

   The information box closes.

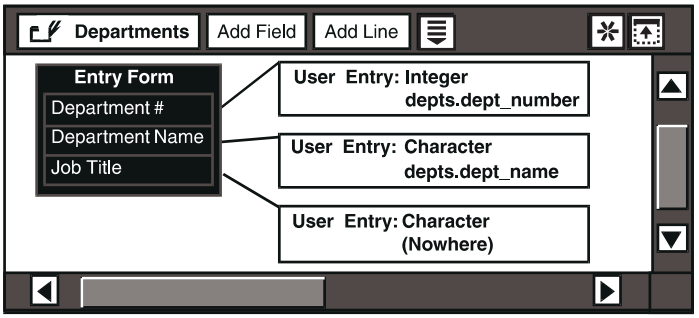
If you double-click on a choice or section field, the heading for choices or sections appears. See Chapter 8, “Designing Choice Fields” on page 40, for information about adding choices to a field, and Chapter 9, “Building Flexibility with the Section Field” on page 54, for information about adding sections to a section field.

Displaying Information for All Fields in a Form

To display database information for all fields in a form:

1. Double-click on the form heading in the main Forms Design window.

   For example, if you double-click on the heading of the Entry Form depiction, the information boxes appear for all the fields in that form.



2. Select the **Entry Form** heading again.

   The information boxes close.

Working with Capsule Applications

When you create a Data Entry icon for use inside a capsule application:

You can automatically enter values into the**@-variable** fields in the User Input Control window of the capsule tool.

You can verify that the Data Entry icon is set to store data in *@-variables*.

You can create a Data Entry icon to automatically open as soon as a capsule window opens.

However you decide to use your Data Entry icon with your capsule application, it is a good idea to leave the Forms Design icon in the capsule icon along with the Data Entry icon. You can move the Forms Design icon way to the bottom or the right so that it is not visible. Leaving the Forms Design icon in the capsule icon makes it easy to update the Data Entry icon later.

The following sections describe these tasks in greater detail. For information about using a Data Entry icon inside a capsule application, refer to *Data Entry User's Guide*.

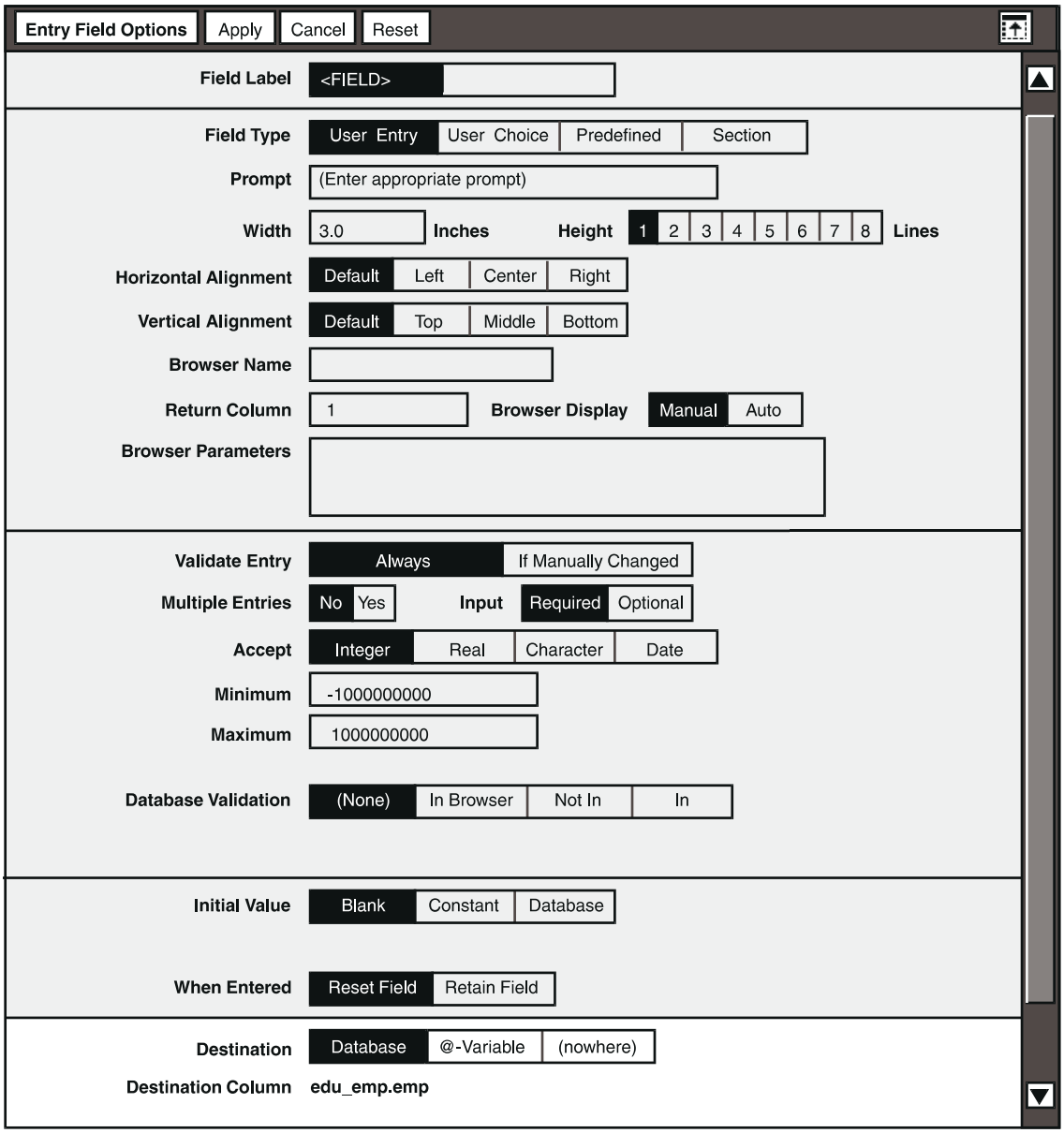
Creating a Form to Set @-Variables

You can customize a Data Entry icon to set *@-variables* inside a capsule application. To do so, create a Data Entry icon with fields whose destinations are specific *@-variables*. The Data Entry icon then automatically enters the information into the **@-variable** fields in the capsule tool's User Input Control window.

To create a field with data stored in *@-variables*:

1. Select a field in the **Entry Form** depiction.
2. Open the Entry Field Options window.

   The Entry Field Options window appears similar to the following illustration.



3. Select **@-Variable** in the **Destination** field at the bottom of the window.

   Because **@-Variable** is the default, it might already be selected.

4. Enter a value in the **@-Variable** field. For example, enter **@A**.

5. Close the Entry Field Options window.
6. Repeat this procedure for each field that you want to store in an *@-variable*, assigning a different *@-variable* for each field.

Verifying @-Variable Entries

Verify your *@-variable* declarations by testing with a Data Entry icon. Enter values into the Data Entry form, then check inside the User Input Control window to see if the values have been updated in the **@-variable** fields.

To open the capsule tool's User Input Control window, click on the **Show Controls** button in the capsule window header.

The User Input Control window is shown in the following illustration.

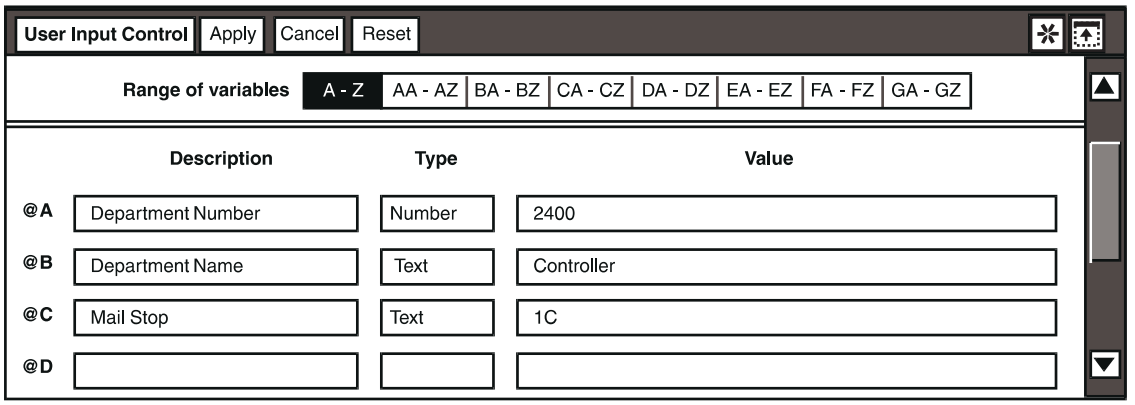


Figure 26. User Input Control window

Opening a Data Entry Icon Automatically

You might want to ensure that the Data Entry icon opens before running the capsule application. To rename a Data Entry icon so that it opens automatically:

1. Select the Data Entry icon.
2. Open the Icon Options window.
3. Delete the old name, and type Auto Open in the **Name** field.

4. Close the Icon Options window and deselect the Auto Open icon before you run the capsule application.

The location of the Auto Open icon is not important. It can be placed anywhere inside the capsule window. However, only one Auto Open icon can automatically open in a capsule window. If you insert two such icons, the first one you inserted automatically opens when you open the capsule window.

Creating a Forms Design Icon from a Data Entry Icon

You might want to use the same Forms Design icon to create other similar Data Entry icons. Therefore, you should retain the Forms Design icon associated with each Data Entry icon that you create. However, if you inadvertently delete a Forms Design icon, there is a way to recreate it from its Data Entry icon. With the new Forms Design icon, you can:

Replace the original Forms Design icon

Redesign the Data Entry icon that you are presently using

This procedure works only if you are logged on to the same desktop that you used to create the Data Entry icon.

To recreate a Forms Design icon:

1. Log on to the desktop that created the Data Entry icon.
2. Open the Data Entry icon.
3. Select **Create Forms Design Icon** in the Data Entry window's Special menu.

   The pointer changes, and the following message appears on your desktop:

   Select location of new Forms Design icon.
4. Select an empty location on your desktop or in a container.

   The Forms Design icon appears on your desktop.

Generating Post-Entry SQL

Generate post-entry SQL when you want to run one or more SQL statements after each transaction. Post-entry SQL enables you to:

Simultaneously insert or update data in more than one table

Run an SQL program that exists in another document or in the Forms Design tool

Produce an audit trail

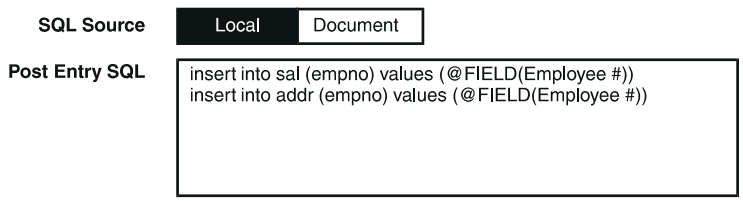
Construct constraints on data

You can specify SQL statements in either the Update Form Options windows. Because the SQL statement is associated with the form, you can specify different SQL commands for each form.

Choose from two methods of specifying post-entry SQL:

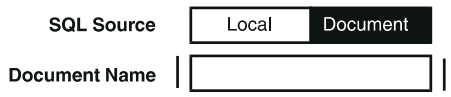
Locally

Select **Local** in the **SQL Source** field and enter statements in the **Post-Entry SQL** field.



Remotely

Select **Document** in the **SQL Source** field and name the location of the statements in the **Document Name** field.



If you are entering the SQL statements locally, be sure to separate each command line by pressing Shift+Enter on your keyboard. This procedure ensures that all the commands are run before access of the tables is allowed.

If you are inserting or updating data in multiple tables, see the section “Inserting or Updating Data in More Than One Table.”

If you are entering the SQL statements remotely, see the section “Accessing SQL Statements in a Source Document” on page 122 for more information.

As with the SQL Entry tool, you can incorporate flow control and conditional processing by including &statements, functions, and function statements in the post-entry SQL. See Appendix A, “Identifying Keywords and Statements” on page 145 for a list of the &statements and functions that are supported by the Forms Design tool. See the section “Constructing an SQL Constraint” on page 123 in this chapter for information about the &CONSTRAINT function.

Inserting or Updating Data in More Than One Table

You can insert or update data in more than one table by specifying post-entry SQL statements. You write these statements in the **Post-Entry SQL** field of the form options window, or refer to the document that contains the SQL statements.

For example, you can write statements that insert rows into the Salary and Name and Address tables for a specific employee. The statements appear similar to the following example:

```
insert into sal (empno) values (@FIELD(Employee#))
insert into addr (empno) values (@FIELD(Employee #))
```

The @FIELD keywords in the above example reference the **Employee #** field in the Entry form.

Set the destination of the column to **None** or to **@-Variables**.

See the section “Discovering the @-Keywords” on page 145 in Appendix A for information.

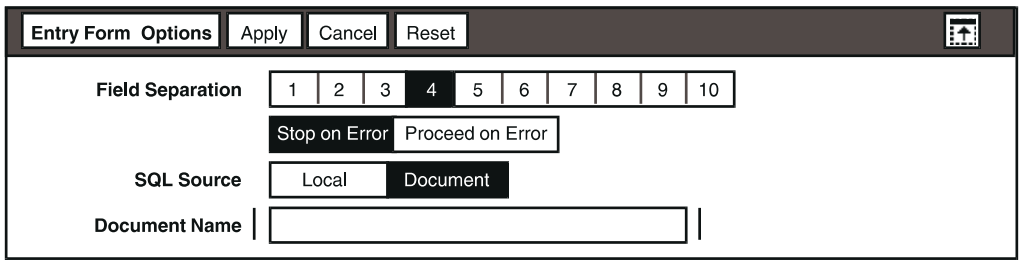
Accessing SQL Statements in a Source Document

You can use existing SQL statements in a source document rather than enter SQL statements into the **Post-Entry SQL** field.

To retrieve SQL statements from a source document:

1. Select **Document** in the **SQL Source** field.

   The **Document Name** field appears as shown in the following illustration.



2. Enter the name of the document that contains the SQL statements.

The following methods are used to locate a source document:

If the form is in a container, such as in a capsule application, the container is searched. If the source document is not found within the container, then the desktop is searched. If still not found, then System|File Drawers is searched.

If the form resides on the desktop, the desktop is searched. If the source document is not found, then System|File Drawers is searched.

The Data Entry tool can locate a source document that is placed in a folder or a file drawer if a path name is included in the Entry Form Options window. This method is useful if your source documents are filed in a central location for easy maintenance.

Make sure that any file drawer you specify in a path name is on the system disk of your file server. For example, if you specify SQL Script #1 as your source document, and your source documents are filed in the SQL Scripts folder within the Data Access Libraries file drawer, enter Data Access Libraries|SQL Scripts|SQL Script #1 as a path name in the **Document Name** field. These containers are then searched for the

source document. Produce the vertical bar in the path name with the broken vertical bar key on your keyboard.

If the source document cannot be found, the following error message is returned:

Cannot find the Post-Entry SQL source document.

Refer to *SQL Entry User's Guide* for more information on using a source document.

Producing an Audit Trail

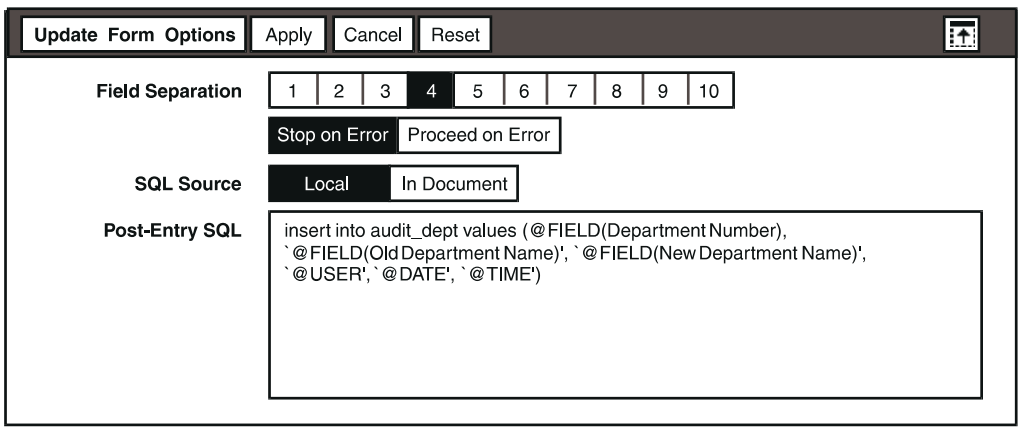
You can produce an audit trail through the **Post-Entry SQL** field if you have defined the proper database tables. The audit trail lets you keep track of database activities.

To produce an audit trail:

1. Select the heading of a form in the main Forms Design window.

   For example, select the **Update Form** depiction.

2. Open the Update Form Options window.



This illustration shows the post-entry SQL associated with the Update Form depiction. The Update Form is being used to change the department name in a Departments table. The post-entry SQL statements insert a row into an audit table that keeps track of the changes made to the Departments table.

Refer to *SQL Entry User's Guide* for information on SQL statements.

Constructing an SQL Constraint

You can use the &CONSTRAINT function to construct an SQL constraint for a field. By using &CONSTRAINT, you can create an SQL statement that:

Allows for multiple data items in a field – for example, 0, 1, or more.

Specifies the type of column to be constrained.

Combines &CONSTRAINT( ) with the rest of the SQL statement by using terms such as WHERE, AND, IN, and =.

The SQL statement must be in the following format when using the &CONSTRAINT function:

select *columnname1* from *tablename* &CONSTRAINT(*columnname2*, *datatype*, *operatortype*, *datavalues*)

where:

columnname1	is the name of the database column that contains the values used for validating the field.
columnname2	is the name of the database column that contains the result.
tablename	is the name of the database table that contains the values used for validating the field.
datatype	specifies the type of data to be constrained. The data type must be in uppercase. Valid types include:
CHAR	is a value in a character format.
NUMBER	is a value in a numeric format.
DATE	is a value converted to the appropriate server date.
operatortype	is either EQUAL or NOTEQUAL.
datavalues	are the single or multiple values to be used as the constraint values. Quotes are not used for text.

Table 21 lists three examples of SQL statements that use the &CONSTRAINT function. The examples are based on the following SQL statement:

select Sales from Salestable &CONSTRAINT(City, CHAR, EQUAL, @FIELD(City))

Table 21. Examples using &CONSTRAINT in SQL statements

If the value is	The SQL statement is
City=NY	Select Sales from Salestable where city = NY'
City=NY, CA, GA	Select Sales from Salestable where city in ('NY', 'CA', 'GA')
No values in field	Select Sales from Salestable

Be sure to include any additional clauses before the &CONSTRAINT to prevent parsing errors.

See Appendix A, “Identifying Keywords and Statements” on page 145, for information about other available &functions.

Specifying Forms Design @-Keywords

You can enter @-keywords in a field to generate system data, such as the current date, or in reference fields within Data Entry forms.

For example, when you create a Data Entry icon from a Forms Design icon, the name of that Data Entry icon is the same as the Forms Design icon, and also includes the current date. The @-keywords @ICON and @DATE were specified in the Forms Design icon so that the Data Entry icon is named in this manner.

To assign @-keywords to name a Data Entry icon:

1. Open the Forms Design icon.
2. Select **Show Data Entry Options** in the Special menu.

   The Data Entry Options window appears as shown in Figure 27.

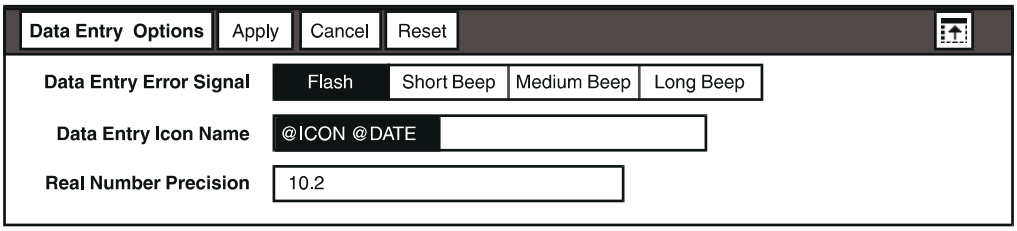


Figure 27. Data Entry Options window with @-keywords

3. Enter @-keywords in the **Data Entry Icon Name** field.

   In this example, @ICON and @DATE appear in the **Data Entry Icon Name** field. When a Data Entry icon is first created, it automatically assumes the Forms Design icon name and current date.
4. Close the Data Entry Options window when complete.

The procedure is the same for entering any @-keyword into a field. The only @-keyword that has more than one function is @FIELD when it is used with the GLOBAL feature. See the section “Assigning a Global Name” in this chapter for information about global names.

Assigning a Global Name

The GLOBAL feature assigns a global name to a field so that it can be referenced from any other field in a form. This feature is particularly helpful if there are many dependent fields and duplicate field labels in a form. Refer to the global name with the @FIELD @-keyword in the **Post-Entry SQL** field of the Update Form Options windows. The field with the global name does not have to be visible in the Data Entry window when post-entry SQL is executed.

For example, if you name a field Product Name GLOBAL:Prod12, the global name is Prod12. You can then refer to that name in the **Post-Entry SQL** field with

@FIELD(GLOBAL:Prod12). Although you see the entire field name in the Forms Design icon, only Product Name appears in the Data Entry icon.

Setting Up Conditional Expressions with @PARMs

When working with browse lists, you can access different parts of the browse list from different fields. The @-keywords @PARM1, @PARM2, ...@PARM10 allow you to set up a constraint without specifying the conditions in the **General Qualifier** field. The specific conditions for each field are then given in the **Browser Parameters** field of the field options window. The position of the condition in the **Browser Parameters** field relates to the number of the @PARM. All fields that use this browse list must follow the constraint pattern given in the **General Qualifier** field.

For example, the expression where @PARM1 = @PARM2 when given in the **General Qualifier** field can be filled by "City, Atlanta" in the **Browser Parameters** field. This expression, when compared to where @PARM1 = @PARM2, translates to City=Atlanta. The **@PARM** relationship is further explored in Table 22.

Table 22. @PARM relationships

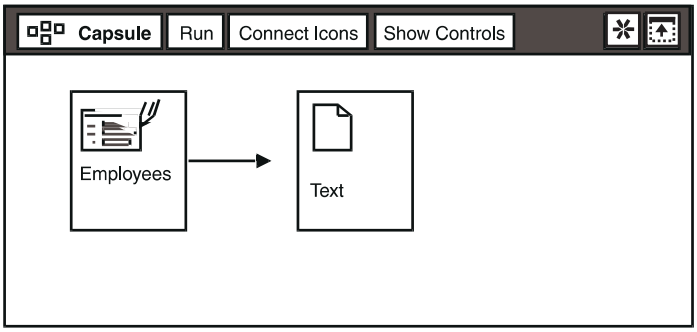
General Qualifier field of the Browser Options window	Browser Parameters field of the field options windows	Resulting SQL
where @PARM 1 = @PARM 2	City, 'Atlanta'	City = 'Atlanta'
where @PARM 3 < @PARM 4	City, 'Atlanta', Employee #, 2365	Employee # < 2365
where @PARM 5 > @PARM 6	City, Atlanta', Employee #, 2365, JOB CODE, 9011	JOB CODE > 9011

You can use the @PARMs in either the **General Qualifier** field, the **SQL Text** field in the Browser Options window, or the **Formula** field in the Browser Options window. The expression entered in any of these fields finds the values in the **Browser Parameters** field in the field options windows.

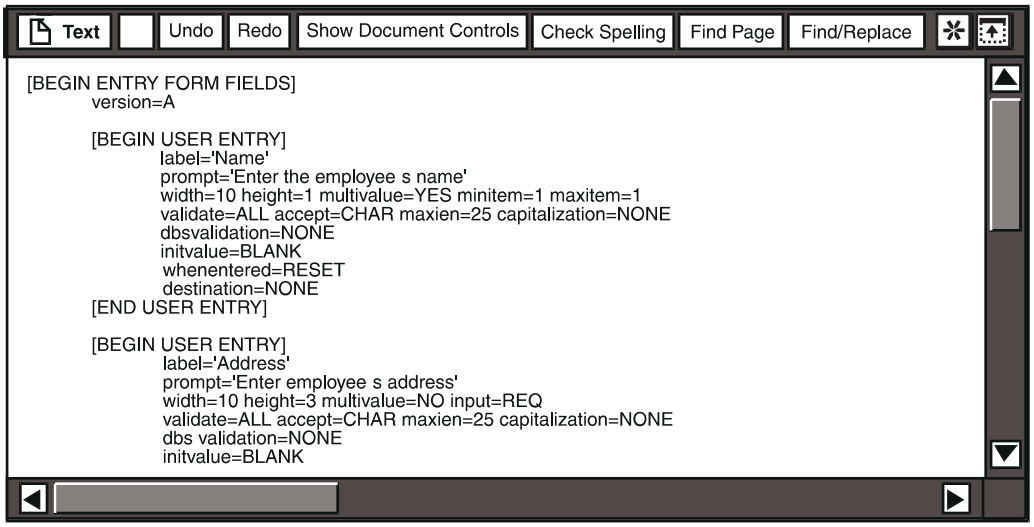
Modifying Forms Design Tools with Scripting

Sometimes it is necessary to modify information within a Forms Design tool. One way to change the information is to find the appropriate field and make a different selection. But what if you need to change something that appears throughout the Forms Design tool – for example, the name of a field?

With scripting, you use a Text tool to find and replace the information that needs to be modified. Scripting works by connecting the Forms Design icon to a Text icon within a capsule window. The following illustration shows the connection necessary to create a script from a Forms Design icon.



As shown by the above illustration, the Forms Design tool acts as the source of information, and the Text tool as the destination. When the capsule application runs, the Text tool appears as shown in the following illustration.



Make any necessary changes to the script within the Text tool.

Before you make any changes to the Text tool, see the section “Knowing the Acceptable Values for Scripting Keywords” on page 147 in Appendix A.

If a form is large and you do not want to create one document, you can create documents for different portions of the Forms Design icon by defining a region name in the arrow that connects the Forms Design and Text icons. Use the names of the forms for regions – for example, set the source region to entry form or entry. By using this method, you can break down the Forms Design script into three Text icons: one each for the entry, retrieval, and update forms.

When you have made all necessary changes, you can create a Forms Design tool from your modified Text tool. Place the modified Text tool into a capsule window and connect it to a blank Forms Design icon as shown below.

Attention: Do not connect the Text tool to your original Forms Design tool. This process clears all existing fields and definitions from the Forms Design tool, and then constructs a new Forms Design tool. If an error is made within the Text tool, you will have lost the original, working Forms Design tool.

For information about the Text tool, refer to the IBM IDS *Text User's Guide*.

Using Column and Table Name Indirection

You can re-direct a Forms Design tool to another table without changing all of the occurrences of the table name in the icon by using the WTDD. This feature saves time and reduces error by enabling you to make the needed change in the WTDD, rather than in the Forms Design tool.

For example, use indirection if you have created a Forms Design icon for one department, and another department wants to use it for their databases. Instead of updating all of the occurrences of the table name, use the DBA tool to change the internal names in the WTDD so they match the internal names in the original table (check first to be sure that other applications are not using the same table). You can then configure the Forms Design icon to the new category in the Icon Options window, and it will reference the correct physical table and columns.

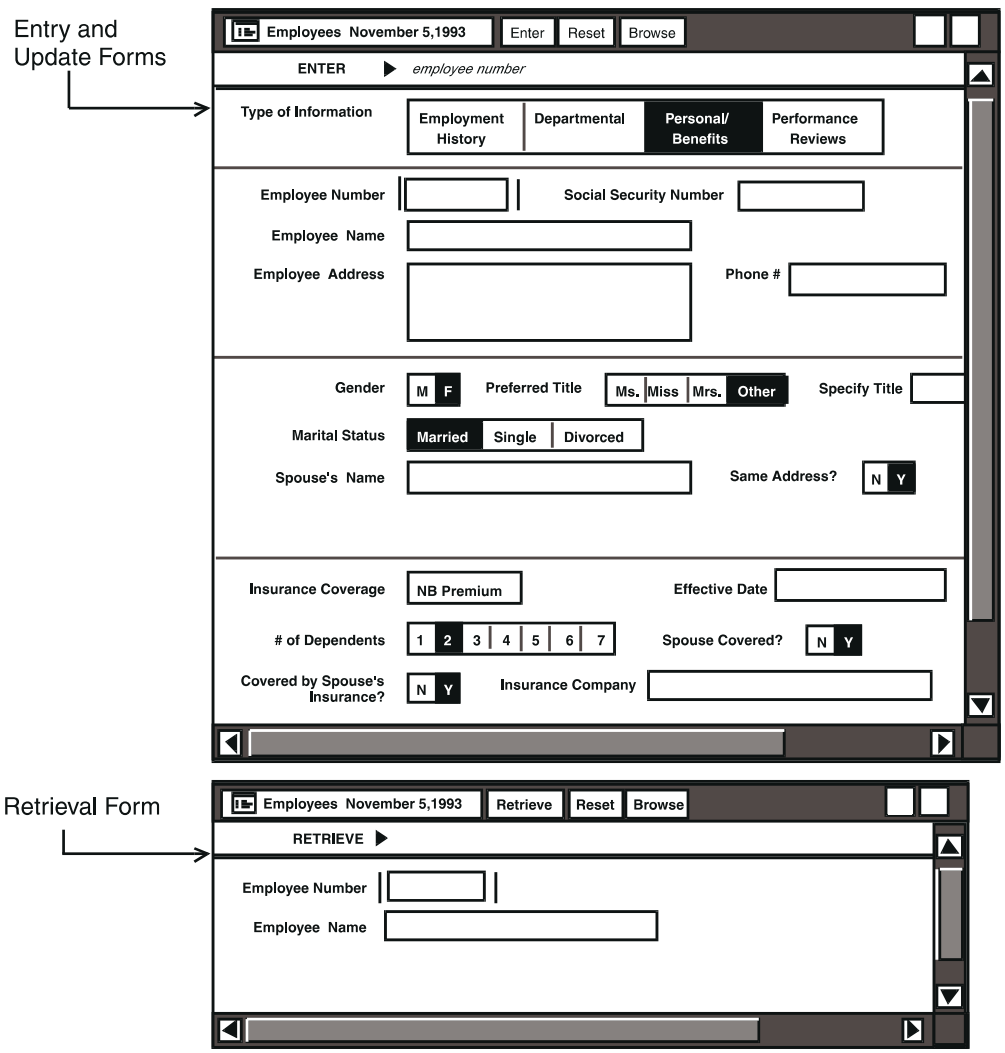
Also use indirection when you migrate tables from one database to another; for example, DB2 to Sybase. Copy the WTDD to the new database, change the SQL names for tables and columns, and then make all icons reference the new database.

Refer to the IBM IDS *DBA Tool User's Guide* for more information about changing physical names in the WTDD.

Chapter 16. Designing Complex Forms

Now you have learned all of the individual benefits of Forms Design's features. But how are all of these features put together to create a form? This chapter provides a fairly complex form, and explains how certain aspects of the design were accomplished. References to more detailed information in the guide is also provided.

Suppose you are requested to create a Data Entry icon for Personnel to enter employee information into their databases. Because they want to update information later, you must create all three forms: entry, retrieval and update.



© Copyright IBM Corp. 1996

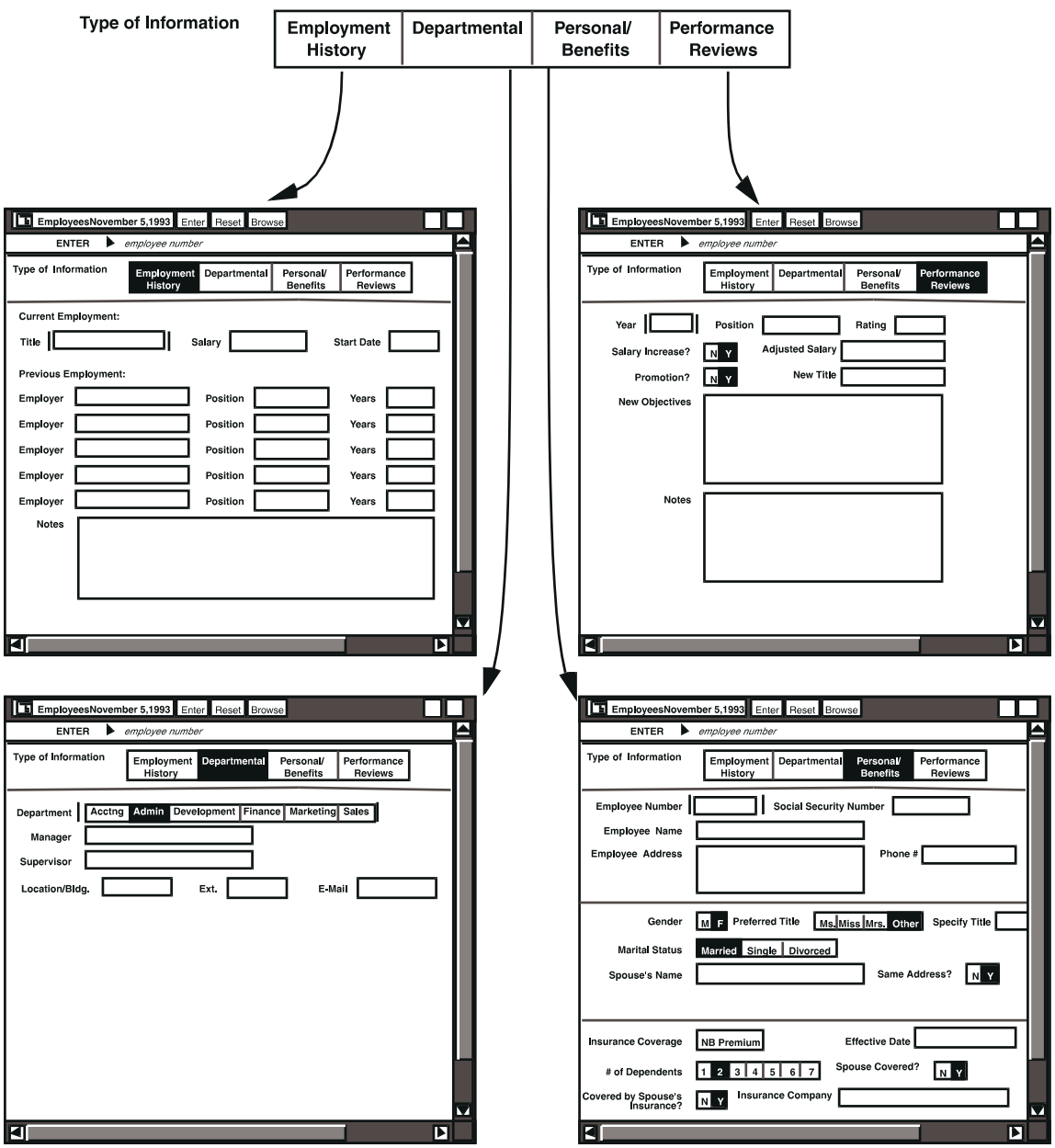
The above illustration only shows one section of a section field for the entry and update forms, but the logical structure of the forms is apparent.

While the entry and update forms contain all related fields, it is only necessary for the retrieval form to contain fields for the employee number and name. All of the associated information will be retrieved so that it can be updated or deleted.

The rest of this chapter focuses on the structure of the sections of the **Type of Information** field for the entry and update forms, and the contents of the retrieval form.

Structuring the Entry and Update Forms

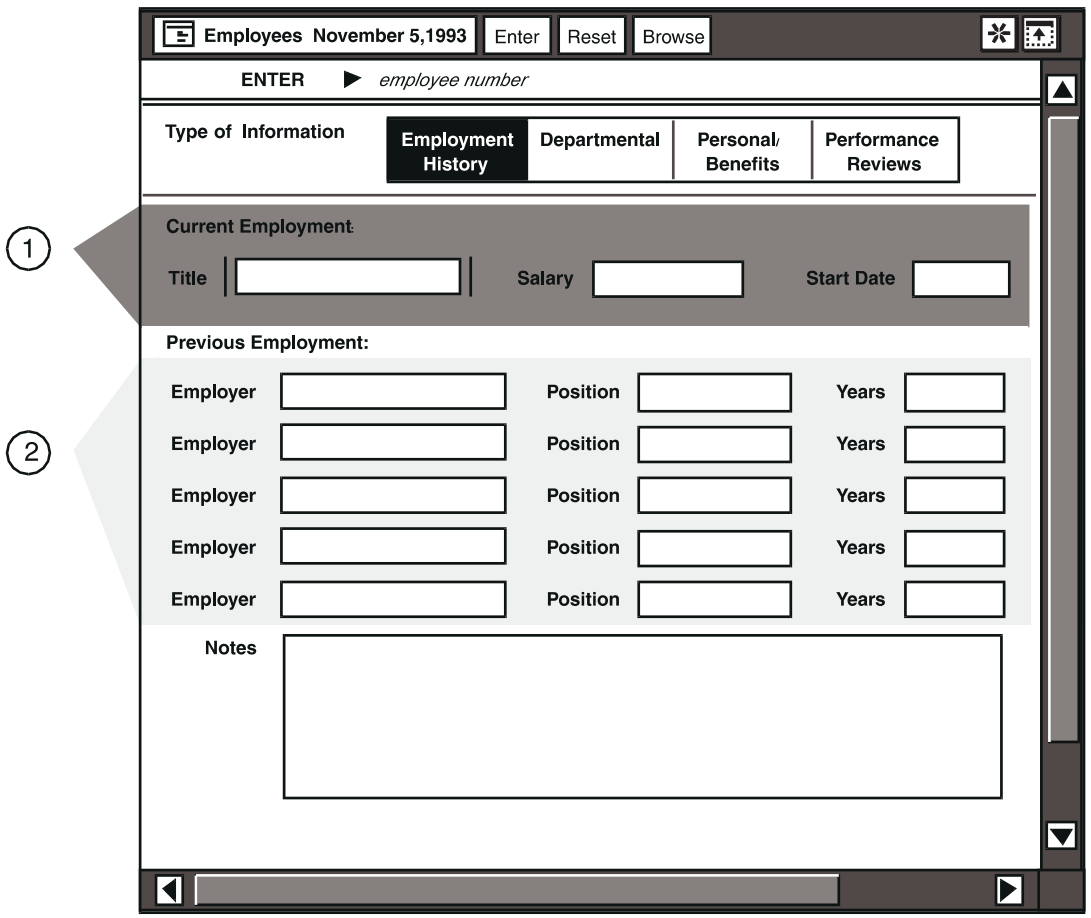
The overall structure is accomplished by one feature: the section field.



By using the section field, you can divide information into conceptual segments, and display it in different screens. The user enters information into all fields dependent upon

a section field, and then validates the entries all at once by clicking on the **Enter** button. However, unless you define it to be so, the user does not have to enter information into all the dependent fields.

Looking at the Employment History Section



Placing fields side-by-side helps to use space effectively. The **Title**, **Salary** and **Start Date** fields are part of a group in which the field layout is horizontal and the field size variable. These options are set in the Group Options window. See the section “Grouping Fields” on page 106.

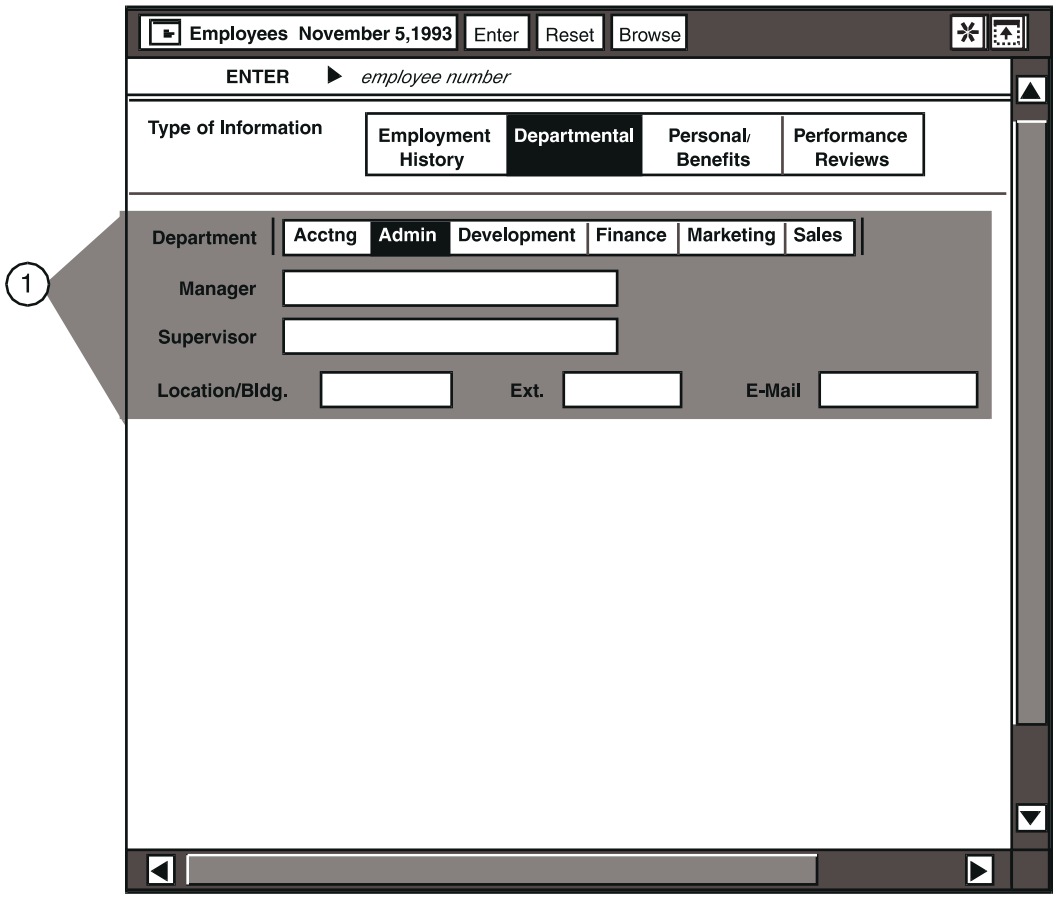
Making fields align horizontally and vertically gives your form an organized appearance. To align fields as shown in the Previous Employment section you must use the grouping feature.

To accomplish the alignment shown in the Previous Employment section, you must create four groups: one group for each of the columns, and then one group to group all the columns together.

When grouping a column's fields, specify vertical alignment so that the fields align in a column. When grouping the columns together, specify horizontal alignment so that the column groups align by rows.

You can also use this method to align fields of differing lengths.

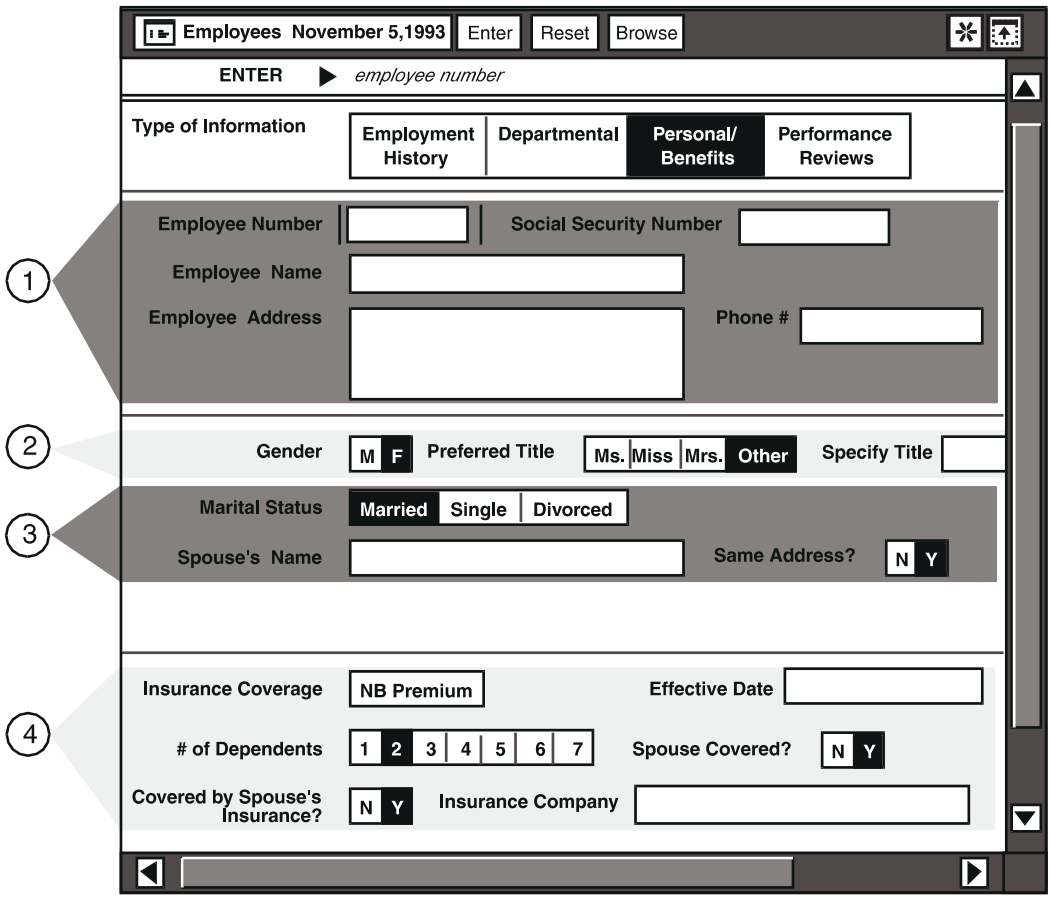
Browsing Through the Departmental Section



You can create a browse list to locate the name of the employee's manager and supervisor, and to find the employee's location in the company. Just create one browse list with these columns: Manager, Supervisor, Location/Bldg., Ext., and E-Mail. Then set the Return Column in the Entry Field Options window to the correct column. See Chapter 13, “Working with the Browse Feature” on page 90, for information about constructing browse lists and setting the **Return Column** field for each column.

You can make the browse list even more efficient by using the **Department** field to constrain the information retrieved into your browse list while leaving the columns the same. For example, if **Marketing** is chosen from the **Department** field, then the browse list only returns information for the Marketing department. Enter a constraint into the Browser Options window to constrain by department. To enter this constraint, you must select **Custom** from the **Browser SQL** field, and type your constraint in the **General Qualifier** field. For example, to constrain the information in the browse list by the department chosen in the **Departments** field, enter 'DEPT='@FIELD(Department)' in the **General Qualifier** field. See the section “Determining the Browser SQL” on page 94 for more information.

Exploring the Personal/Benefits Section



1. As shown in the first example, placing fields side-by-side helps to use space effectively. The fields in this section are part of groups in which the field layout is horizontal and the field size variable. These options are set within the Group Options window. See the section “Grouping Fields” on page 106 for more information.

Be sure to allow enough space in fields for the required information to be entered. For example, the **Phone #** field is 1.5 inches wide so that a full phone number with area code can be entered. Also, use patterns when necessary as in the **Social Security Number** field.

To ensure that an employee number is not duplicated, validate the **Employee Number** field against the database. See the section “Preventing Duplicated Data” on page 72 for information.

2. Again there are fields placed side-by-side, but the structure here is accomplished through dependencies between the fields. The selections within the **Preferred Title** field are dependent upon the selection of **F** in the **Gender** field. For a male, the **Preferred Title** field might appear as in the following illustration.

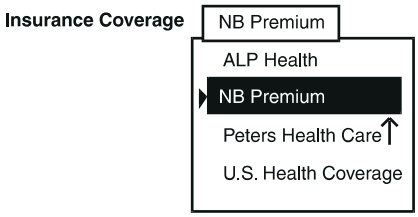
Furthermore, the **Preferred Title** field enables the user to enter a title not given as a choice by providing the **Other** choice. The **Specify Title** field accommodates those choices of which the designer was not aware.

The structure and placement of these fields is not accomplished by grouping, but is achieved with fields and dependent fields. The **Preferred Title** field is actually two different fields: one with choices for a female, another with choices for a male. The **Specify Title** field is also two different fields. The dependent fields of both the **Gender** and **Preferred Title** fields are placed to the right of the fields from within the field options window. See the sections “Layering with Choice Fields” on page 44, “Creating Dependent Fields” on page 45, and “Positioning Dependent Fields” on page 52 for information.

Eliminate work for the user by providing only those fields that are necessary. For example, the **Spouse's Name** and **Same Address?** fields only appear because **Married** is selected from the **Marital Status** field. Furthermore, an address field for the spouse does not appear on the form unless **N** is selected from the **Same Address?** field because it is not necessary. Create this structure by using fields and defining dependent fields.

Along the same lines, the entire section called out by point 4 can be made dependent upon the selection in the **Marital Status** field. For example, a married employee needs to be asked if her spouse is covered by insurance, but a single person does not. This format requires more fine-tuning on the part of the forms designer, but is rewarding for the user. See the sections “Layering with Choice Fields” on page 44 and “Creating Dependent Fields” on page 45 for more information.

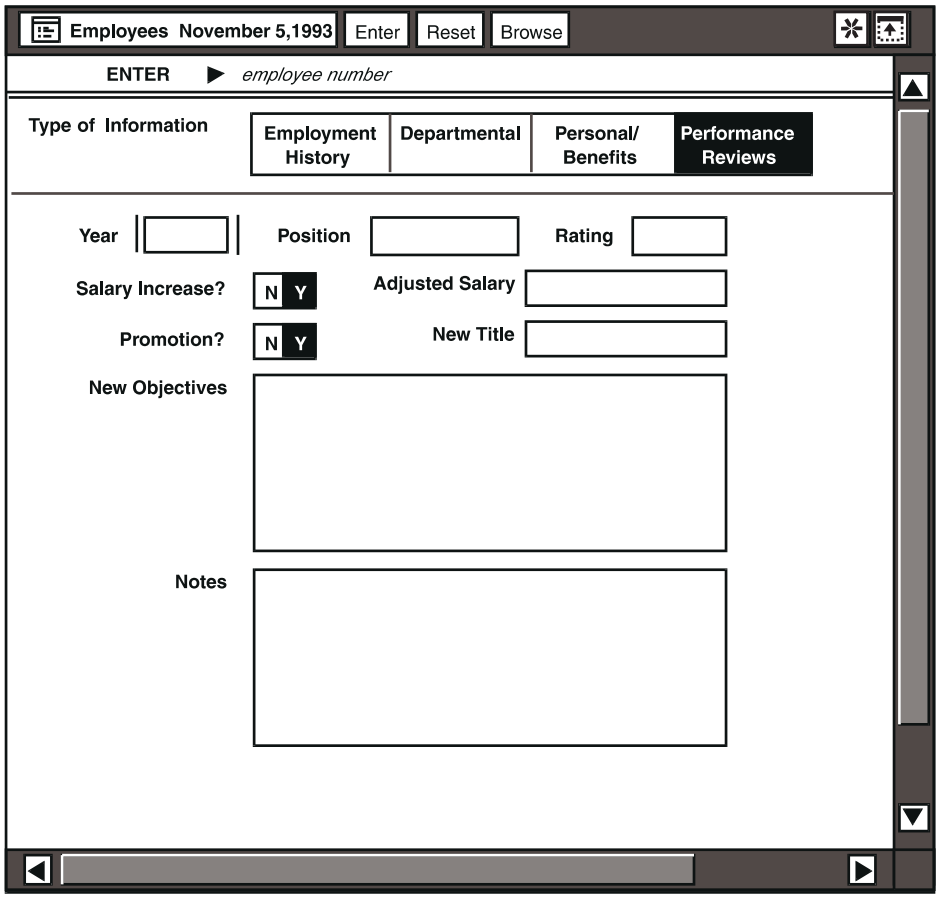
Place choices in a pull-down menu when there are many choices that take up too much space to display. An example is the **Insurance Coverage** field. Take a look at what that pull-down menu holds:



Keeping those choices in a pull-down menu saves a lot of form space. See the section “Presenting Choices” on page 48 for information.

Reviewing with the Performance Review Section

This example uses several of the features described by the previous sections.



The features used in this example include:

Placing fields side-by-side through grouping.

Creating hierarchies of fields through choices and dependent fields.

Structuring information so that fields only appear when necessary; for example, as a result of a choice.

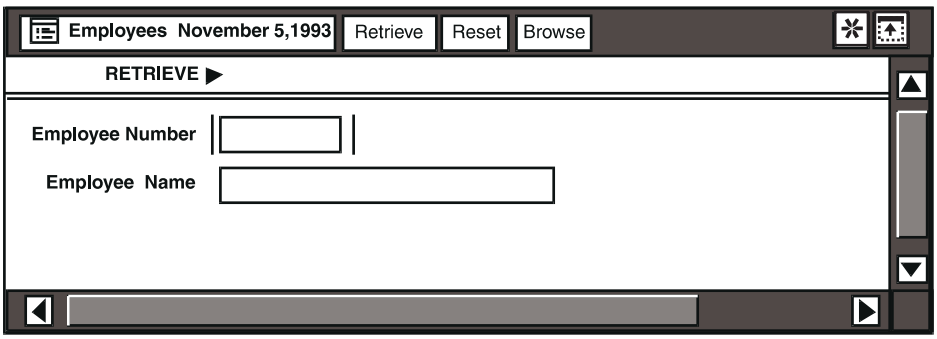
Aligning fields horizontally and vertically through grouping.

Using one browse list for several fields.

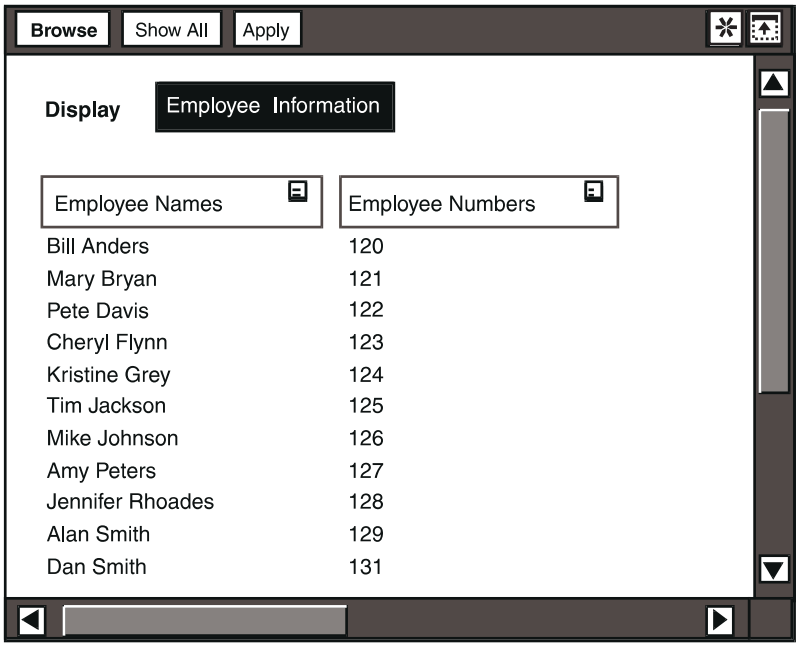
A pull-down menu could also have been used for the **Rating** field if there are only a few levels of scoring.

Creating the Retrieval Form

The retrieval form only needs to contain fields to retrieve the employee number and name. When designing these fields, it is important to ensure the accuracy of entries.



For both of these fields, the entry should be chosen from a browse list. You can create one browse list to be used by both fields. The browse list might appear as shown in the following illustration.



Set the return column within the field options window to the appropriate column. See the section “Figuring Out the Return Column” on page 37. See Chapter 13, “Working with the Browse Feature” on page 90, for information about setting up browse lists.

Be sure to validate entries against the browse list as well. See the section “Validating Data with a Browse List” on page 72 for information.

Chapter 17. Improving Security and Performance

Due to the complex nature of the Forms Design tool, there are many security and performance issues you must take into consideration when designing forms. These issues relate to tasks such as creating physical tables for the users, defining those tables to the logical database, specifying the users who can access those tables, and monitoring the performance. You might need the help of your database administrator to perform these tasks. This chapter contains information related to these DBA activities as they pertain to the Forms Design and Data Entry tools, including:

- Implementing database security
- Determining SQL performance
- Executing SQL statements through the Forms Design window
- Using nonproduction tables

Beyond this chapter, refer to *Administering Databases with IDS*, *System Administration Guide and Reference*, or *Working with the Intelligent Decision Server* for more information on assigning access keys, predefining access to the database tables, and using the **Secured** field.

Implementing Database Security

There are four methods with which you can limit user access in the data entry process and, consequently, ensure the security of the database:

- Assign user IDs and passwords
- Predefine user access
- Assign access keys
- Predefine database table access

Each of these methods is discussed in the following sections.

Assigning User IDs and Passwords

Configure the Forms Design icon with the user's database name and password. When you create Data Entry icons from that Forms Design icon, the Data Entry icons automatically have the correct configuration. If you configure the Forms Design icon with your user ID and password, you might forget to change that information when you create the Data Entry icon.

Predefining User Access

When you design forms, you control the data that a user can enter, retrieve, delete, or update. For example, suppose you are designing a form to allow a user to retrieve information from an Employees table in the database. To restrict retrieval to only specific rows from a database table, define a field in the Retrieval Form depiction as predefined, and give it a constant initial value. For example, when defining a **Department**

© Copyright IBM Corp. 1996

Number field, specify a particular department number so that the user can retrieve only the rows for employees in that specific department.

You also can allow the user to retrieve rows for more than one department. Either define a field whose values are validated against specific database values, or a field whose choices are the specific department numbers.

To constrain two or more fields where there is a dependency between the constraints, you must set up a dependency between those fields. For example, assume that you want to enable the user to retrieve all rows from an Employees table for people who work for division 00A or 00B. Assume that division 00A has departments 10, 20, and 30, and division 00B has departments 10, 15, and 20. With the proper SQL statement, you can set up a browse list to limit the division and department numbers, and indicate that values must be in the browse list. Or, you can restrict a user to certain data using dependent fields.

To restrict a user with dependent fields:

1. Define a field and set up the choices.

   For example, using a field named **Division ID**, define the choices as **00A** and **00B**.

2. Define dependent fields for each of the choices.

   For example, define a **Department Number** field as a dependent of 00A, and another **Department Number** field as a dependent of 00B.

3. Define the choices for the dependent field.

   For example, the **Department Number** field dependent on choice 00A has choices 10, 20, and 30. The dependent field for choice 00B has choices 10, 15, and 20.

When the user selects **00A** in the **Division ID** field, the **Department Number** field appears as shown in the following illustration.

Figure 28. Forms Design Log and SQL Controls windows

When the user selects **00B** in the **Division ID** field, the **Department Number** field appears as shown in the following illustration.

Regardless of how you design the forms, the DBA is still responsible for granting access to the database tables.

See Chapter 8, “Designing Choice Fields” on page 40, for instructions on how to create dependent fields.

Assigning Access Keys

By assigning an access key, you identify a user or a group of users to a Data Entry icon, and restrict the user's ability to view or change the icon's contents. An access key is a number from 1 to 64 assigned by the system administrator.

Because a user can mail icons to other desktops, there is still potential for a security violation. Even if an icon has an access key, the receiver of the icon becomes the owner and is therefore able to use it. You might want to discuss predefining database table access with the DBA. See the following section, “Predefining Database Table Access with the Secure Feature” for information.

Predefining Database Table Access with the Secure Feature

The DBA is responsible for controlling access to the database tables by creating users and granting them privileges. You, as the forms designer, can ensure that only those desktop names that are granted database access by the DBA can use the Data Entry icon to access the tables in the database.

See the section “Securing a Forms Design Icon” on page 8 for information about securing an icon.

Determining Data Entry SQL Performance

The way SQL statements are used has an effect on the performance of the Data Entry tool. You can create fields that are:

Database-defined

Database-validated

Database-initialized

Generating Database-defined Choice Fields

If you generate choices for a field from the database, the SQL statement that generates the choices is executed when the Data Entry icon is opened.

If the field is large, it could take some time to execute that SQL statement. If delay in opening the Data Entry icon becomes a problem, it is better to make the field a field whose data is validated against the database.

Generating Database-validated Entry Fields

When you define a field, there are several ways you can validate the values in that field against the database. Each of the following options have different implications for database performance. You can specify field values:

In a browse list

In a database table

Not in a database table

Specifying Field Values in a Browse List

When you define a field whose values are specified in a browse list:

The browse list is generated as specified in the Browser Options window.

The user can always update the browse list. This option should be used with browse lists that are fairly static.

If the browse list is fairly dynamic, you might want to update the values automatically. There are three methods with which you can automatically update browse lists:

Each time the browse list is used for validation

With this method, the browse list is not stored with the Data Entry icon. Instead, the list is generated each time a field is validated. This choice should be used with fields that are constantly changing.

Each time the Data Entry icon is opened

With this method, the icon then takes longer to open.

Each time the database is queried

With this method, browse lists are updated only when necessary because they are only updated when a row is returned that differs from one that is specified in the Browser Options window.

Specify the frequency with which a browse list is updated in the Browser Options window. See Chapter 13, “Working with the Browse Feature” on page 90, for information on browse lists.

Specifying Field Values in, or Not in, a Database Table

You can specify that an entry should either be in, or not in, a database table. The Data Entry icon runs an SQL statement each time the user enters data to validate the entry. If the database table has a large number of rows, this method could have a significant impact on performance.

Generating Database-initialized Fields

When you define a field that is initialized from the database:

An SQL statement generates the initial value when the Data Entry form is displayed.

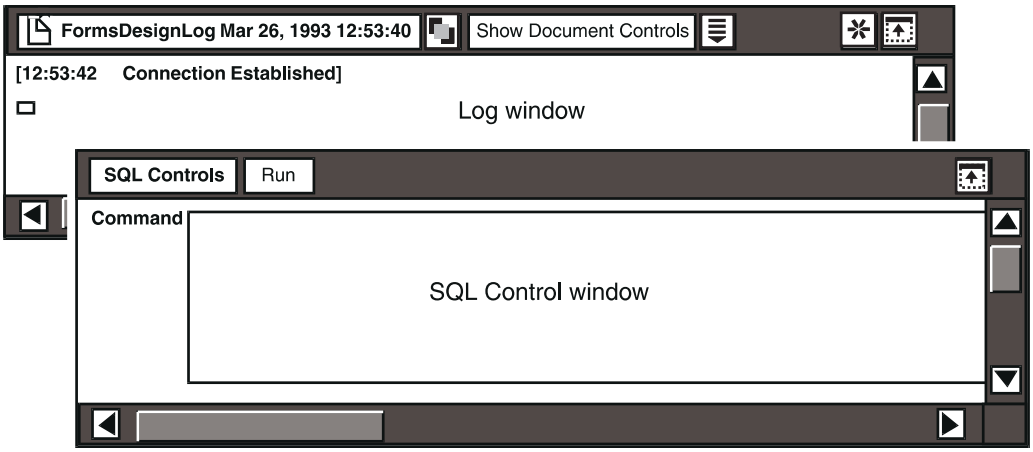
If the field is defined to reset after an entry, the SQL statement is executed each time the user clicks on the **Enter** button in the Data Entry window header.

Executing Commands in the SQL Control Window

Execute SQL statements by entering them through the SQL Controls window. This method is beneficial because you do not need to open an SQL Entry tool each time you need access to the database server.

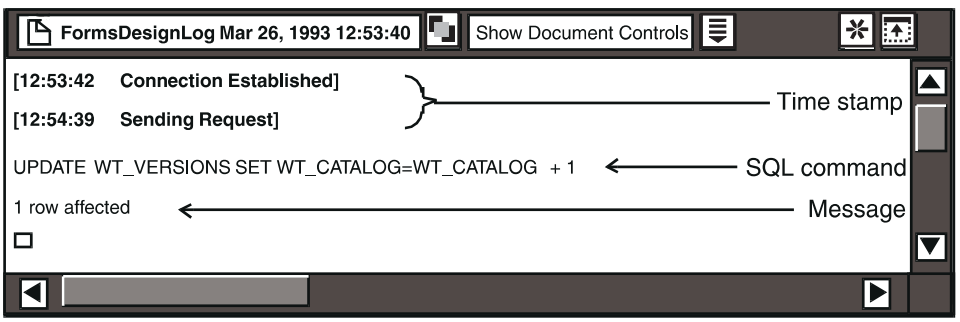
To execute SQL statements using the SQL Controls window:

1. Select Forms Design Log window, shown in Figure 28 on page 140. The log window contains a date stamp that indicates the date and time the capture window was opened. The function of the log window is to capture the SQL activities.



2. Enter or copy an SQL statement into the command area of the SQL Controls window.
3. Click on the SQL Controls window header.

   The Forms Design tool sends the statement to the database server, and two lines are returned to the log window as shown in the following illustration.



After the time stamp is the SQL statement that was sent to the database server, and a message indicating the result of the activity.

When you close the SQL Controls window, the log window closes and remains on your desktop as a record of the SQL activity. A new log window appears each time you open the SQL Controls window.

Working with Production Tables

Production tables, or database tables that hold vital data an organization uses for essential workstation work, must not be impacted by queries. For example, a query issued by a brand manager should not cause the production applications to slow to a crawl and interfere with the schedules of many other people.

When a user enters information, the SQL statement issued by the Data Entry tool is sent to the database server immediately. This immediate update can potentially cause performance or database integrity problems if users are performing queries against the same tables into which other users are updating or inserting data.

One way to avoid potential performance and integrity problems is to adhere to the following guidelines:

Have the users entering data use a nonproduction copy of the tables.

Put the nonproduction tables in a physical database different from the production database.

The users can then apply any changes to the nonproduction copy of the tables. At a later time, you can run the appropriate SQL program to apply those changes to the production tables.

Appendix A. Identifying Keywords and Statements

There are many keywords and statements that assist you when using Forms Design's more technical features. For example, you can specify many @keywords and &statements that are useful when constructing post-entry SQL statements. You can also modify the contents of a script if you know the acceptable values for the keywords. This appendix provides the following:

Definitions of @-keywords

Definitions and syntax for &statements

Definitions and acceptable values for scripting keywords

Discovering the @-Keywords

You can enter @-keywords in a field to generate system data, such as the current date, or reference fields in the Data Entry forms. You must enter all @-keywords in uppercase. Table 23 describes the @-keywords that you can use with the Forms Design tool.

Table 23 (Page 1 of 2). Forms Design @-keywords

@-keyword	Description
@COLUMN	A column in the database. This @-keyword is used only in the SQL statement that initializes a field in an Update Form. You use @COLUMN instead of @FIELD when you want to initialize a field to a value in the database, rather than the current value in the field.
@DATE	The current date.
@FIELD	The contents of the field specified in parentheses at the time the @FIELD is evaluated. The evaluation depends on the context in which it is used. If you use @FIELD in the **Post-Entry SQL** field, the field specified must be visible in the Data Entry window when the post-entry SQL is executed. If there is more than one visible field with the same field name, the one closest to the top of the form is used. If you want to reference a field that goes across subforms, then assign a global name to the field. If you use @FIELD in an Entry Field Options window and the field containing the @FIELD reference is in a path in a hierarchy, the Data Entry icon searches that path, from bottom to top, and uses the first field or choice encountered whose name is specified in the @FIELD. If there is more than one field with that name at the same level in the hierarchy, the Data Entry icon uses the topmost field.
@FIELD (GLOBAL:)	The global name in a field that enables you to refer to any other field within the form. When you use @FIELD with a global name, you must first assign a global name to a field, and then use the same global name in the @FIELD command. In the label area of the field options window, you assign a global name by typing GLOBAL: followed by a name. If the Data Entry tool detects more than one field in a form with the same global name, you receive a message indicating which field contains the conflicting global name. See the section “Assigning a Global Name” on page 125 for more information.
@ICON	The name of the icon.

© Copyright IBM Corp. 1996

Table 23 (Page 2 of 2). Forms Design @-keywords

@-keyword	Description
@PARM(1-10)	A keyword used to set up a constraint on a browse list. The conditions of the constraint, however, are not specified in the expression. The @PARMs allow specification of the conditions from the **Browser Parameters** field in the field options windows. See the section "Setting Up Conditional Expressions with @PARMs" on page 126 for more information.
@TIME	The time in hours, minutes and seconds. (HH:MM:SS)
@USER	The name of the desktop that contains the icon. The desktop name is capitalized exactly as entered by the user when logging on to the desktop.
@USERID	The unique 8-digit hexadecimal number that identifies the desktop.

Learning About &Statements

As with the SQL Entry tool, you can incorporate flow control and conditional processing by including &statements, functions, and function statements in post-entry SQL. Table 24 explains the &statements and functions that are supported by the Forms Design tool

The &DISPLAY statement is not supported in the **Post-Entry SQL** field.

For detailed information on these statements and functions refer to *SQL Entry User's Guide*.

Table 24 (Page 1 of 2). Acceptable functions and &statements in post-entry SQL

Statement/ Function	Syntax	Description
&CONSTRAINT	&CONSTRAINT (column name, data type, operator type, data value)	A function that allows you to constrain a field that accepts multiple data items. For more information on the &CONSTRAINT function, see Chapter 10, "Validating Data" on page 62.
&DATE	&DATE (date)	A function used in an INSERT command and a WHERE clause which allows you to convert date strings to those represented internally by the database server.
&IMPMSG	&IMPMSG ('string')	A function that allows you to post an important message in a Data Entry form. You can specify text and *@-variables* in the message parameter. When the message is displayed by the Data Entry tool, it is preceded by the phrase *Application Message:* to distinguish it from system generated important messages.
&ERROR	&ERROR action ;	A statement that allows you to control the method of handling errors (with the exception of parsing errors) by specifying an action such as STOP, IGNORE, or MESSAGE.
&GOTO	&GOTO label ;	A statement that directs the flow of a program to the SQL statement after the specified & label variable.

Table 24 (Page 2 of 2). Acceptable functions and &statements in post-entry SQL

Statement/ Function	Syntax	Description
&IF	&IF condition THEN GOTO label ;	A statement that uses a condition to direct the flow of a program to the SQL statement after the specified & label variable.
&label	& label :;	A statement that uses an ampersand (&) to precede a word of your choice in SQL, and allows you to direct the flow of the program when used with the &GOTO statement.
&SET	&SET @variable = expression ;	A statement that allows you to set *@-variables* while executing a program.
&SQLNAME	&SQLNAME (tablename [.columnname])	A statement that allows you to take advantage of table name or column name indirection. The table name parameter is the internal table name specified in the WTDD.
COLVAL(n)	COLVAL(n)	A function that accesses the results of the last-executed statement using the value from the *n*th column of the first row of data.
ERRNUM	ERRNUM	A function that returns the error number of the last SQL statement executed from the database.
NROWS	NROWS	A function that returns an integer indicating the number of rows affected by the last statement executed.

In addition, the **Post-Entry SQL** field also supports @-keywords, as described in the section “Discovering the @-Keywords” on page 145 in this chapter.

Knowing the Acceptable Values for Scripting Keywords

Certain keywords can only be followed by one of several predefined values. These predefined values correspond to the values displayed through the graphical user interface (GUI). Table 25 explains the predefined values for the scripting language.

Table 25 (Page 1 of 3). Acceptable values for certain scripting language keywords

Keyword	Function	Allowed values
error	Determines the data entry error signal.	“FLASH”, “SHORT”, “MEDIUM”, “LONG”
updatebrowserchoice	Defines how to update the browse list.	“REQUEST”, “OPEN”, “VALIDATION”, “SQL”
browserSQLchoice	Specifies the source of browser SQL.	“LOCAL”, “DOCUMENT”, “CUSTOM”
sqlsource	Specifies the source of post-entry SQL.	“LOCAL”, “DOCUMENT”
onerror	Determines the type of action taken when an error occurs.	“STOP”, “PROCEED”

Table 25 (Page 2 of 3). Acceptable values for certain scripting language keywords

Keyword	Function	Allowed values
width	Sets the width of an field.	Any number that correlates to ISUs (Incredibly Small Units). The width is represented by ISUs in the code regardless of whether the width is set in inches or centimeters. There are 57,600 ISUs per inch, and 145,728 per centimeter.
multivalue	Specifies if multiple values are allowed.	“NO”, “YES”
input	Determines if input is required for the field.	“REQUIRED”, “OPTIONAL”
initvalue (except update form)	Sets the initial value for a field.	“BLANK”, “CONSTANT”, “DBS”, “TODAY”, “NOW”, “USER”
initvalue (update form)	Sets the initial value for a field.	“COLUMN”, “CONSTANT”, “DBS”, “TODAY”, “NOW”, “USER”
choicewidth	Specifies if the choice width is constant or variable.	“CONSTANT”, “VARIABLE”
choicedef	Specifies the source of the choice definition.	“PREDEFINED”, “DBS”
destination	Determines the storage of information.	“DBS”, “VARIABLE”, “NONE”
accept	Specifies the acceptable data types.	“INTEGER”, “REAL”, “CHARACTER”, “DATE”, “TIME”, “TIMESTAMP”
date resolution	Sets the date resolution for a field or a browser column.	“DAY”, “WEEK”, “QUADWEEK”, “MONTH”, “ODDBIMONTH”, “EVENBIMONTH”, “QUARTER”, “YEAR”
capitalization	Determines the capitalization of entered character strings.	“NONE”, “ALL”, “INITIAL”
validate	Specifies the validation method of input.	“ALWAYS”, “IFCHANGED”
compute	Computes the aggregate value in SQL to determine the initial value of the field.	“MIN”, “MAX”, “COUNT”, “AVG”, “MAX+1”, “OTHER”
whenentered	Determines what to do with a field value when a user clicks on the **Enter** button.	“RESET”, “RETAIN”
dbsvalidation	Sets the type of DBS validation to be performed on a field.	“NONE”, “BROWSER”, “NOTIN”, “IN”
stovalue	Specifies what value is stored for section or fields.	“NAME”, “VALUE”, “DBS”

Table 25 (Page 3 of 3). Acceptable values for certain scripting language keywords

Keyword	Function	Allowed values
match	Matches the value for the destination description in section or fields.	“NAME”, “VALUE”
operatormay	Determines permission to delete rows associated with retrieval form options.	”NOTDELETEROWS”, “DELETEROWS”
initvalue	Sets the initial value for choice or section fields in an update form.	“COLUMN”, “DBS”, “OTHER”
length (browser column)	Specifies the length of a displayed browser column.	“SHORT”, “LONG”
order (browser column)	Sets the sorting order for browser columns.	“ASCENDING”, “DESCENDING”, “OTHER_COLUMN”
display (browser column)	Controls the display of a browser column.	“SHOW”, “HIDE”
displayas	Sets the data type of the displayed values in a browser column.	“INTEGER”, “REAL”, “CHARACTER”, “DATE”, “TIME”, “TIMESTAMP”
precision (browser column)	Determines the precision for a column that contains real numbers in a browse list.	“0000.1”, “000.02”, “00.003”, “0.0004”, “.00005”
fldlayout	Sets the field layout type for a group.	“VERTICAL”, “HORIZONTAL”
fldsize	Specifies that the field size for fields within a group are either constant or variable.	“CONSTANT”, “VARIABLE”
sepunits	Sets the separation units used in vertical field separation.	“PIXELS”, “LINES”
horizontalalign	Determines the horizontal alignment for a group relative to its space.	“DEFAULT”, “LEFT”, “CENTER”, “RIGHT”
verticalalign	Determines the vertical alignment for a group relative to its space.	“DEFAULT”, “TOP”, “MIDDLE”, “BOTTOM”
browserdispmode	Specifies when a browse list is displayed.	“MANUAL”, “AUTO”
choicestyle	Sets the style of choice or section fields to be either displayed horizontally or in a pull-down menu.	“FLAT”, “MENU”
placement	Determines placement of dependent fields for a choice or section field.	“BELOW”, “TO_THE_RIGHT”
initvalue	Sets the initial value for a choice or section field.	“LEFTMOST”, “DBS”, “OTHER”

Glossary

access key. A code number that restricts access to a specific icon. Only users whose desktops include the code number can use the icon. See *key holder*.

access privileges. A set of permissions assigned by an icon's owner to control who may view or change the icon.

Add Browser. The window header button that adds a browse list to the Browser Library header.

Add Choice. The window header button that adds a choice to a list of predefined choices.

Add Comment. The window header button that adds a field in which a forms designer can type instructions or comments.

Add Field. The window header button that adds a field to an entry form, a retrieval form, or an update form depiction.

Add Group. The window header button that adds a group to an entry form, a retrieval form, or an update form depiction. See *Group*.

Add Line. The window header button that inserts a line in a form.

Add Section. The window header button that adds a choice to a section field.

Apply. A window header button that puts into effect any changes that the user has specified in an options or control window.

@-keyword. A system-defined variable that returns current values within a Capsule application or within the headers and footers of a printable icon.

@time. A system-defined @-keyword that returns the day of the week, date, and time.

@user. A system-defined @-keyword that returns the user's desktop name.

@-value. The value assigned to an *@-variable* in the Capsule tool's User Input Controls window. See *@-variable*.

@-variable. A control variable that can be used to represent a value in any tool that can be run inside a Capsule application. Therefore, these values are used as constraints in the Forms Design and Data Entry tools. See *@-value*.

Browse. A button that appears in the window header of the Data Entry tool when applicable to a particular field. The Browse button opens a window that displays a browse list.

browse list. A list displayed in a window that opens when the data entry operator clicks on a Browse button or when the data entry operator selects in a field that is set up to automatically open a browse list. The browse list contains all acceptable entries for each field associated with the *Browse* button.

browser. A browse list as defined in the Forms Design icon by the forms designer.

Browser Column Options. The window in which the forms designer sets options that apply to the browse list column.

Browser Library. The heading in the main window which defines the browsers associated with the Data Entry icon.

Browser Options. The window in which the forms designer sets options for the browse list.

Cancel. A window header button that closes an options or control window without saving any changes other than those that were saved the last time the user clicked on the **Apply** button.

Capsule tool. A tool in which the user builds and performs application programs with other tools.

capture window. A Text window that records the SQL statements sent to the database from the Command area of the SQL Controls window.

catalog. Tables, maintained by the database manager, that contain descriptions of objects, such as tables, views, and indexes.

category window. The window that displays a specific category of database tables for which an icon is configured.

© Copyright IBM Corp. 1996

choice number. The relative number of the choice within an option. The first choice has a choice number of one.

Choice Options. The window in which the forms designer assigns a name and value to a choice in a predefined field.

choice value. A notation used to represent a Choice Name. The Choice Value can be stored in the database instead of the Choice Name.

constraint. A factor that limits the range of data retrieved from the database to reduce the amount of unneeded data that is returned. For example, the forms designer might constrain data based on territory and period, so that only data relevant to New York during March, 1992 is retrieved.

container. Any folder, file drawer, or other icon that can contain other icons.

Create Column. A button in the main window header that adds a customized column to the browser.

Create Data Entry Icon. The choice in the Special menu that creates a Data Entry icon based on the forms in the main window.

Create Forms Design Icon. The choice in the Special menu of the Data Entry tool that recreates a Forms Design icon from a Data Entry icon.

database validation. The process by which the value entered into a field is compared with values in the database.

Data Entry Options. The window where the forms designer or the data entry operator set options that apply to the Data Entry icon.

Data Entry Tool. The tool that is used to enter, retrieve, delete, and update data on a database server, or to set *@-variables* in the User Input Control window of a Capsule application.

data type. The internal representation of data in a physical database column. Data types generally available in the IDS logical tables are integer, real, character, and date. Other data types available in some IDS logical tables include null, long varchar, varchar, time, and timestamp.

decimal precision. The number of displayed decimal places in a number.

default value. A value that is assumed when no value has been specified.

dependent field. A field that is displayed only when a particular choice is selected in a choice or section field.

depiction. The illustration of an object. For example, the categories in the Forms Design category window, or the forms in the main Forms Design window, or an icon within a Capsule window.

displayed precision. A number that has been rounded or truncated. When a number has been rounded off, the displayed number rounds down if the last digit is less than 5, and round up if the last digit is 5 or greater (vice versa for negative numbers). The number is truncated at the specified number of decimal places.

entry form. The name of the form used to enter information into a database table or a Capsule application.

field type. The method of data entry in a form. For example, a choice field type allows the data entry operator to make a selection from a group of several choices. Other field types include entry, predefined, and section.

file drawer. A container, created or renamed by the system administrator, that is shared among desktops. A file drawer typically contains folders. See also *folder*.

foreign key. A column in a database table that references a primary key in another database table. A foreign key can be viewed in the Forms Design tool's category window.

form. A component of the Data Entry tool used to either enter, retrieve, delete or update information in a database, or supply values to the User Input Controls window of a Capsule application.

forms designer. The person who creates Data Entry icons using the Forms Design tool.

Forms Design Log. The window that displays a record of all actions performed by the forms designer while the main Forms Design window is open.

function keys. Programmed keys on the workstation keyboard used to manipulate icons and windows.

group. A collection of fields that may be placed horizontally or vertically.

Group Options. The window in which the forms designer specifies the options for groups.

information box. A box that appears in the Forms Design window when the forms designer double-clicks on a form depiction field or heading. If the forms designer selects a field, this box displays the field type, the acceptable data type, and the name of the database column to which the field corresponds. If the forms designer selects the depiction heading, the information box displays information about all fields in the depiction.

initialization. The process of giving value to a data object at the beginning of its lifetime.

initialization area. An area in the Field Options window that contains information about how the field is initialized.

Line Options. The window in which the forms designer sets options for lines in a Data Entry form.

list separator. A character, such as a comma, used to break a text string into separate items.

operator. A symbol the user selects or enters along with a value to define the amount, range, or characteristics of data to be retrieved from a database.

options window. Any window that the user can open by using the Options key or the options function on the pop-up function pad. Options windows enable the user to view and modify the properties of the selected object.

parser. Software that analyzes the syntax and determines the meaning of SQL statements.

path name. A designation for the location of a file or icon. For example, a path name for a file on a computer disk specifies the names of the disk and all the directories leading to the file; a path name for a script called by a Cooperative Application Services transformer specifies the name of any Capsule applications in which the script is located; the path name for an OS/2 icon specifies the names of any OS/2 folders that must be opened to get to the icon.

pattern. A series of characters arranged to illustrate the correct format for entering character data into a field. For example, 999-99-9999 is a character pattern for a Social Security number. The forms designer can define a pattern that the Data Entry tool uses to validate character data that a data entry operator enters.

physical database. The physical set of stored data tables that underlies a logical database

physical table. A table in a physical database.

pixel. An acronym for picture (pix) element (el), the smallest element on a video display screen to which a color or light intensity can be assigned.

post-entry SQL. One or more SQL statements that run each time a forms designer enters data into an entry form or update form.

predefined field. The field in a form that contains information the data entry operator cannot change.

primary key. One or more columns in a database table that uniquely identifies the rows in that table. The primary key can be viewed in the Forms Design tool's category window.

production tables. Tables against which users are currently and actively issuing queries. For example, tables used as a high access on-line transaction system.

relational data. In a relational database, information that is related to other information and whose relationships can be defined in the form of tables consisting of rows and columns of data.

restricted menu item. A choice in the Special menu that can be used only by the system administrator or forms designer.

retrieval form. The form used to retrieve information from a database table.

Retrieval Form Options. The window in which the forms designer can define the appearance and operation of the retrieval form.

section field. The field in a form that displays several choices from which the data entry operator can make a selection. Each choice displays a section that contains dependent fields. All fields within each section are active and are validated and entered when the data entry operator clicks on the **Enter** button.

Section Options. The window that displays the section field options.

secure. To restrict access to an icon.

Special button. A button in some window headers and on the far-right side of the message area which opens the Special menu.

Special menu. A menu that appears when the user points to the Special button and presses the left mouse button. In tool windows, the menu choices are usually

reserved for special functions or controls associated with the icon.

SQL. Structured Query Language.

SQL Control. The window in which the forms designer enters SQL statements and they are sent to the database.

Structured Query Language (SQL). A command language for use with relational databases. The language consists of statements to insert, update, delete, query, and protect data. IDS tools use Structured Query Language to communicate with database servers.

storage area. An area in the Field Options window where the forms designer specifies where data is stored.

system administrator. The person who is responsible for operating and configuring IDS, including its workstations, servers, and network.

table. A single database file, consisting of data organized in columns and rows.

table name. The title or label for a division of a database category.

update form. A form used to modify information in a database table.

Update Form Options. The window in which the forms designer sets options to define the appearance and operation of the update form.

User Input Control window. The window in a Capsule tool that is used to store *@-variables*.

user name. A name used to access a desktop, system, gateway, or database service.

validation. The checking of entered data for completeness, accuracy, or reasonableness.

validation area. An area in the Field Options window used to specify how data is validated and how character strings are edited.

view. A copy of a query, which can be named, stored, edited and redisplayed for later use.

Workstation Tools Data Dictionary (WTDD). A logical data dictionary that helps standardize the workstation view of data. The WTDD stores the external names of data items and defines how the items are to be displayed. The WTDD is implemented as a set of standard relational tables, so it can be managed through icon-based workstation tools.

WTDD. See *Workstation Tools Data Dictionary*.

IDS Publications

This section lists IDS publications, including order numbers. To order copies of the books listed here, or to get more information about a book, see your IBM representative.

General Publications

Getting Started with the Intelligent Decision Server (SC26-8892)

Working with the Intelligent Decision Server (SC26-8893)

Installation, Configuration, and Administration Publications

Installing IDS LAN Components (GC26-8877)

System Administration Guide and Reference (SC26-8890)

Messages and Codes (GC26-8878)

Database Gateway Services Publications

Installing Database Gateway Services for UNIX Environments (GC26-8884)

Configuring and Administering Database Gateway Services (GC26-8885)

Administering Databases with IDS (SC26-8891)

Host Services Publications

Planning MVS Host-to-LAN Connectivity (GC26-8880)

Installing Host Services and Configuring Host-to-LAN Communications for MVS (GC26-8881)

Administering Host Services for MVS: DB2 and Cooperative Application Services (GC26-8882)

Developing Applications with Cooperative Application Services (GC26-8883)

Configuring and Administering Mail Gateway Services (GC26-8923)

Open Data Access Service Publications

Developing Applications with Open Data Access Service (SC26-8888)

Data Access Tools Publications

Browser User's Guide (SC26-8900)

Data Entry User's Guide (SC26-8897)

DBA Tool User's Guide (SC26-8894)

SQL Entry User's Guide (SC26-8898)

Data Analysis Tools Publications

BASIC Tool User's Guide (SC26-8908)

Query User's Guide (SC26-8901)

Spreadsheet User's Guide (SC26-8904)

Presentation Tools Publications

Layout User's Guide (SC26-8902)

Plot User's Guide (SC26-8905)

Reporter User's Guide (SC26-8903)

Text User's Guide (SC26-8896)

Application Development Tools Publications

Transformers Guide (SC26-8907)

Capsule User's Guide (SC26-8906)

Forms Design User's Guide (SC26-8899)

Interacting with PC Applications (SC26-8895)

Installing and Configuring IDS Open Clients (GC26-8889)

© Copyright IBM Corp. 1996

Index

Special Characters

@-keywords
  @PARMs, uses for 126
  global name, assigning to field with 126
  list of 146
  using in Forms Design 125
@-variables, setting in Data Entry form 117, 119
@COLUMN, definition of 145
@DATE, definition of 145
@FIELD, definition of 146
@ICON, definition of 146
@PARM, definition of 146
@TIME, definition of 146
@USER, definition of 146
@USERID, definition of 146
&-statements, list of 146
&CONSTRAINT function
  definition and syntax of 146
  examples of usage 124
  format of 124
  uses for 123
&DATE, definition and syntax of 146

A

access keys, assigning for security 141
access to database, methods of limiting 139
Add Browser button, uses for 91
Add Choice button, uses for 42
Add Comment button, uses for 11, 113
Add Field button, uses for 11
Add Group button, uses for 12, 106
Add Line button, uses for 11, 111
Add Section button, uses for 55
adding fields to a form, deciding on method 16
aggregates, finding initial value with 79
ascending order, sorting columns into 100
audit trail, producing with SQL 123

B

backslash, in patterns 68
blank
  entries, with N/A values 65
  field
    adding to a form 113
blank *(continued)*
  field *(continued)*
    creating 32
    initializing to 78
browse list
  adding to library 91
  associating with field 37
  automatically displaying 38
  calculations, performing on data 96, 101
  column
    adding with SQL 94, 95
    adding yourself 96
    combining from different tables 96
    constraining data in 95
    copying from Category window 96
    Create Column button, using 96
    customizing yourself 96
    data type, setting 101
    database table, specifying 98
    displaying 101
    hiding 101
    length, determining 99
    naming 98
    not defined in WTDD, creating 96
    number of items within, determining 99
    options for 98
    showing 101
    sorting 100
  constraining with General Qualifier field 134
  constraints, @PARMs, setting up with 126
  converting existing 105
  data type, choosing for columns 96
  displaying 90
  formula, using within 101
  manually displaying 38
  options for 92
  return column, finding 103
  SQL, specifying columns with
    locally 94
    remotely 95
  updating
    based on SQL 93
    dynamic lists 142
    upon user's request 93
    upon validation 93
    when icon is opened 93

© Copyright IBM Corp. 1996

browse list *(continued)*
using one for several fields 102, 104, 133
validating data with 72
Browser Column Options window
created with Create Column button 97
browser library, overview 11
Browser Options window 92
buttons, uses for 11

C

calculations, performing in browse lists 96, 101
capitalization, of character strings 67
capsule application
@-variables
setting in Data Entry form 117, 119
verifying 119
opening Data Entry icon automatically 119
User Input Control window, displaying 119
using Data Entry with 117
Capture SQL item, uses for 12, 142
category window
copying columns from 96
copying fields from 13, 14
foreign key field 50
category, specifying for icon 6
changing Forms Design tools 126, 128
character string
initializing 78
patterns, acceptable formats 67
validating, options for 66
choice field
dependent fields
positioning 52
retrieval form, constraining information 74
storing data
database storage options 87
Choice Options window 42
choice value, for database generated choices 50
choices
creating
for field 41
defining
through database 50
yourself 41
distinct values, generating 50
generating from database 50
naming
field 42
section field 55
choices *(continued)*
presenting
in a field 48
in a section field 58
column
adding
with SQL 94, 95
yourself 96
combining from different tables 96
constraining data in 95
copying from Category window 96
creating with Create Column button 96
customizing yourself 96
data type
choosing 96
setting 101
database table, specifying 98
hiding 101
indirection with WTDD 128
initializing a field to 79
length, determining 99
long 99
naming 98
not defined in WTDD, creating 96
number of items within, determining 99
options for 98
short 99
showing 101
sorting
ascending order 100
by another column 100
descending order 100
commas
interpreted by IDS 64
within entries 64
comments, embedding special 112, 114
configuring a Forms Design icon 5
constant field size, for groups 110
constant value, initializing a field to 79
constraint
calculation, performing on browse list 101
constructing with SQL 123, 124, 134
setting up with @-PARMs 126
converting existing browse lists 105
copying fields
from other fields 15
from the category window 13, 14
Create Column button, uses for 97
Create Data Entry Icon item, uses for 12

creating Data Entry icon 15
customizing columns, for browse lists 96

D

Data Entry icon
creating 15
creating Forms Design icon from 120
Entry Form Options window 19
error procedure 20, 22
error signal 18
field types 28
finding 5
form combinations 19
naming 18
with @-keywords 125
opening automatically in capsule icon 119
real numbers, setting precision 18
redesigning 120
Retrieval Form Options window 21
setting options for 18, 19
Update Form Options window 21
Data Entry Options window 18
data type
character
capitalization 67
length 67
pattern 67, 68
choosing for browse list columns 96, 101
date
acceptable formats 69
creating ranges of 68
precision, setting 68
integer, creating ranges 66
real number, creating ranges 66
setting 65
data values
browse list
calculations, performing on 96
number of in column 99
storing
default storage 89
in @-variables 88
in database 87
nowhere 89
options for 85
data, inserting into more than one table 121
database information
for forms, displaying 117
single fields, displaying 116
Database Name field 6
database name, specifying for icon 6
database security
access keys, assigning 141
defining fields to restrict access 140
methods of ensuring 139
passwords, assigning 139
securing icons 8
database tables
column, specifying for browse list 98
nonproduction, reasons for using 144
production, performance when using 144
database value
initializing a field to 79
initializing with aggregates 79
database, performance issues 141
dates
acceptable formats 69
creating ranges of 68
initializing 78
precision of 68
defaults
initial value 84
storage 89
validation 74
deleting data, giving users permission 21
dependent fields
creating
for fields 45, 47
for section fields 57
layering with 44, 135
positioning
for fields 52
in section fields 59
restricting database access with 140
security, enhancing with 140
storing correct entry 44
validating all at once 54, 132
depictions in Forms Design window, overview 11
descending order, sorting columns into 100
display limits, column exceeding 99
displaying columns, in a browse list 101
duplicating entries 83

E

embedding special instructions 112, 114
entries
data type, setting 65
duplicating 83

entries *(continued)*
multiple entries
allowing 64
conditions of 64
not allowing 65
optional 65
prohibiting duplication of 72
required 65
resetting 83
retaining 83
storing
default storage 89
in @-variables 88
in database 87
nowhere 89
options for 85
timestamping 84
validating 62
all at once 54, 132
always 64
as a character string 73
as entered 73
browse list, with 72
choosing a method 64, 71
default validation 74
for dynamic databases 64
for static databases 64
in database 73
no database validation 71
not in database 72
only if changed 64
process of 64
entry field
aligning horizontally 35
data type
character 78
database information, displaying 116
multiple entries
conditions of 64
Entry Field Options window 24
entry form
@-variables, in capsule applications 117, 119
capsule application
setting up for 117
verifying @-variable declarations 119
columns from one table, importance of 115
field spacing 20
primary key, importance of using 115
setting options for 19
storing data
default storage 89
entry form *(continued)*
storing data *(continued)*
in @-variables 88
in database 87
nowhere 89
options for 85
Entry Form Options window 19
error procedure 20
field spacing 20
post-entry SQL
location of 20
error signal, setting for forms 18

F

field
adding choices to 41
aligning vertically 36
browse list
associating with 37
constraining data in 95, 134
converting existing 105
displaying 90
displaying manually or automatically 38
options for 92
return column 37, 103
see also browse list 90
updating 93
using one for several fields 102, 104, 134
creating 34, 40
data type
character 67, 68, 135
date 68, 69, 78
integer 66
real number 66
setting 65
defining choices
generating through database 50
yourself 41
dependent fields
creating 45, 47
layering with 44, 135
restricting database access with 140
security, enhancing with 140
storing correct entry with 44
differences from field 40
examples of 33
height, setting 49
information box, displaying 116
initial value
blank 79

field *(continued)*
initial value *(continued)*
column 79, 81
constant value 79
database value 79, 81
default 84
leftmost choice 81
methods of initializing 80
now 79
providing 78
specific choice, setting to 82
today 79
user 79
multiple entries
allowing 64
not allowing 65
naming choices 42
optional entries 65
options for 34, 41, 47
overview 28
presenting choices 48
required entries 65
retrieving necessary data 53, 137
setting height and width 35
storing data
default storage 89
generating database value with SQL 87
in @-variables 88
in database 87
nowhere 89
validating entries 62
always 64
as a character string 73
as entered 73
browse list, with 72
choosing a method 64, 71
default validation 74
for dynamic databases 64
for static databases 64
in database 73
no database validation 71
not in database 72
only if changed 64
performance issues 141
process of validation 64
width of choices, setting 49
field options window
displaying 24
initialization area 27
label area 26
field options window *(continued)*
options contained within 26
storage area 27
validation area 27
fields
@-variables
storing data in 118, 119
verifying entries 119
adding to a form
by copying from other forms 15
with the category window 13, 14
browse list, see browse list 90
choice, see field 40
database information, displaying for 116
Database Name 6
database name, specifying for 6
displaying options for 24
duplicating entries 83
entry, see field 33
global name, assigning 126
groups of
adding fields to 107
creating 106, 133, 135
field size, determining 110
naming 108
options for 109
space between 111
space for fields, determining 110
naming 26
predefined, see predefined field 31
prompt message, changing 31
resetting after entries 83
retaining entries 83
section, see section field 54
server, accessing 6
shortcuts to designing 17
side-by-side
examples of 106, 133, 135
within groups 109
SQL ID, specifying for icon 6
stacking, within a group 109, 133
storing data
default storage 89
in @-variables 88
in database 87
nowhere 89
options for 85
types of, differences between 28, 29
foreign key field, definition of 50

form
@-keywords, using 125
adding fields to
by copying from other forms 15
with the Category window 13, 14
audit trail, producing 123
capsule application
setting up for 117
storing data in 117, 119
verifying @-variable declarations 119
columns from one table, importance of 115
combinations of 19
comment, embedding in 112, 114
considerations when designing 115, 129, 138
constraint, constructing 123, 124
database information, displaying 117
entry form
error procedure 20
example of 130
field spacing 20
post-entry SQL, location of 20
setting options for 19
error signal, setting 18
example of 129, 138
field options
displaying 24
types of 26
fields, see fields 28
information box, displaying 117
instructions, embedding 112, 114
lines
changing type of 112
creating 111
uses of 111
naming 18
overview of types 11
post-entry SQL
audit trail, producing 123
constraint, constructing 123, 124
entering locally 121
naming location of 121
source document, naming 122
precision of real numbers, setting for icon 18
primary key, importance of 115
retrieval form
columns from one table, importance of 115
example of 137
field spacing 21
setting options for 21
setting global options for 18, 19
form *(continued)*
shortcuts when designing 16, 17
source document, naming 122
special instructions, adding 112, 114
SQL, see post-entry SQL 122
update form
error procedure 22
example of 130
field spacing 22
post-entry SQL, location of 23
setting options for 21
valid combinations of 19
views, using with caution 115
Forms Design icon
buttons in window header 11
configuring 5, 8
finding 5
limiting user access 139
log, capturing SQL 142
modifying with scripting 126, 128
overview of 1
passwords, assigning 139
performance of 141
redirecting to another table 128
replacing 120
security
access keys, assigning 141
defining fields to restrict access 140
passwords, assigning 139
securing an icon 8
forms designers, tasks required of 3
formula, performing on browse list 101

G

global name, assigning to a field 126
global options, setting for icon 18, 19
Group Options window 107
groups
adding fields to 107
creating 106
examples of 106, 133, 135
field size, determining 110
fields, space between 111
naming 108
options for 109
placing fields within 109
side-by-side fields, creating 109
space between fields 111
space for fields, determining 110

H

height
  choices, setting for 49
  entry fields, setting for 35
  sections, setting for 61
hiding columns, in a browse list 101
hyphens, in patterns 68

I

Icon Options window 6
  fields 6
indirection, using in Forms Design 128
information box
  for form, displaying 117
  single fields, displaying 116
initial value
  default 84
  entry field
    date 78
  field
    character string 78
    database value 81
    leftmost choice 81
    methods of 80
    specific choice, setting to 82
  section field
    database value 81
    leftmost section 81
    methods of 80
    specific section, setting to 82
initialization options, finding 76
instructions, embedding in a form 112, 114
integer values
  creating ranges of 66
items, number of in browse list 99

L

labeling fields 26
length of browse column, determining 99
library of browse lists, overview 11
Line Options window 112
lines
  creating 111
  default line 112
  type of, changing 112
  used in forms 111
log, capturing SQL 142
long column 99

M

menu
  choices, creating of 48
  section choices, creating of 58
migrating
  databases, changing names in WTDD 128
  icon, changing names in WTDD 128
modifying Forms Design tools 126, 128
multiple entries
  conditions of 64
  not allowing 65

N

N/A values, with optional entries 65
naming
  choices for fields 42
  column for browse list 98
  Data Entry icon 18
  fields 26
  global name, assigning 126
  groups 108
  sections for section field 55
nonproduction tables, reasons for using 144
now, initializing a field to 78

O

optional entries
  allowing 65
  with N/A values 65
ordering columns
  ascending order 100
  by another column 100
  descending order 100

P

passwords
  assigning to users 139
  for icon, setting up 8
pattern
  acceptable formats 67
  conditions 68
  enforcing in field 67, 135

performance
browse list
column length, choosing 99
updating methods 142
generating choices from the database 141
in database-validated fields 141
methods of enhancing 141
nonproduction tables, using to enhance 144
SQL, using to enhance 141
placing fields
dependent fields
of fields 52
of section fields 59
entry, aligning 35
groups, within 109
post-entry SQL
audit trail, producing 123
browse list columns, specifying with
locally 94
remotely 95
choices, generating with 52
constraint, constructing 123, 124, 134
creating statements with Query tool 73
database value, generating with 87
distinct values, generating with 52
entering locally 121
executing in SQL Control window 142
global name, assigning to a field with 126
initial value, determining from database 82
initializing fields to database with 142
inserting data in multiple tables with 121
naming location of 20, 23, 121
performance, enhancing with 141
source document of, naming 122
updating browse lists with 94
uses for 120
validating entries with 73
precision of real numbers
browse list column, setting for 101
forms, setting for 18
predefined field
creating 31
database information, displaying 116
information box, displaying 116
initial value, default 84
initializing 78
overview 28
storing data
default storage 89
in @-variables 88
in database 87
predefined field *(continued)*
storing data *(continued)*
nowhere 89
primary key
definition of 50
importance of using 115
production tables, performance when using 144
prompt message, entering 31
pull-down menu
choices, creating for 48
example of 136
section field, creating for 58

Q

Query tool, using to create SQL 73

R

real numbers
creating ranges of 66
setting precision of 18
replacing Forms Design icon 120
required entries 65
resetting fields 83
resolution of date, setting in browse list 101
responsibilities, of the forms designer 3
retaining entries 83
retrieval form
columns from one table, importance of 115
constraining information 74
deleting data, giving permission 21
example of 137
field spacing 21
primary key, importance of using 115
restricting access to database 140
retrieving necessary data 53
setting options for 21
Retrieval Form Options window 21
return column
finding 37, 103

S

scripting
keywords for 149
using 126, 128
section field
creating 55
dependent fields, positioning 59

section field *(continued)*
example of 132
height of choices, setting 61
initial value
column 81
database value 81
default 84
leftmost section 81
methods of initializing 80
specific section, setting to 82
naming sections 55
options for 58
overview 28
presenting choices 58
retrieval form, constraining information with 74
section value, assigning 55
storing data
database storage options 87
default storage 89
generating database value with SQL 87
in @-variables 88
in database 87
nowhere 89
uses for 54
validating sections simultaneously 54
width of choices, setting 60
securing icons 8
security
access keys, assigning 141
defining fields to restrict access 140
methods of ensuring 139
passwords, assigning 139
server, specifying for icon 6
short column 99
Show Category item, uses for 12
Show Data Entry Options item, uses for 12, 18
showing columns, in a browse list 101
side-by-side fields
creating 109
examples of 106, 133, 135
space between 111
sorting columns
ascending order 100
by another column 100
descending order 100
source document
methods of locating 122
naming 122
Special menu, items within 12
SQL ID, specifying for icon 6
SQL, see post-entry SQL 120
stacking fields
space between, setting 111
within a group 109, 133
storing entries
database storage options 87
default storage 89
in @-variables 88
in database 87
nowhere 89
options for 85
stored value, choosing 87
Structured Query Language, see post-entry SQL 120

T

tables
browse list column
combining from different tables 96
specifying for 98
columns from only one, importance of 115
indirection with WTDD 128
inserting data into multiple 121
multiple, using 115
nonproduction, reasons to use 144
primary key, importance of using 115
production, performance when using 144
redirecting a Forms Design icon 128
timestamping entries 84
today, initializing a field to 78

U

update form
columns from one table, importance of 115
error procedure 22
field spacing within 22
initializing fields to column 79
options for 21
post-entry SQL, location of 23
primary key, importance of using 115
storing data
default storage 89
in @-variables 88
in database 87
nowhere 89
options for 85
Update Form Options window 21

user access
access keys, assigning 141
fields, defining to restrict 140
methods of limiting 139
passwords, assigning 139
securing icons 8
user field, see field 28
User Input Control window, displaying 119
user, initializing a field to 78

V

validating entries
always 64
as a character string 73
as entered 73
browse list, with 72
choosing a method 64, 71
default validation 74
for dynamic databases 64
for static databases 64
in database 73
methods of 62
no database validation 71
not in database 72
only if changed 64
process of validation 64
with an SQL statement 73
values
browse list, number of in column 99
character
capitalization, determining 67
length, setting 67
pattern 67, 68
data type, setting 65
date
acceptable formats 69
creating ranges of 68
precision, setting 68
distinct, generating 50
integer, creating ranges of 66
real number, creating ranges of 66
storing
default storage 89
in @-variables 88
in database 87
nowhere 89
options for 85
variable field size, for groups 110
views, using with caution 115

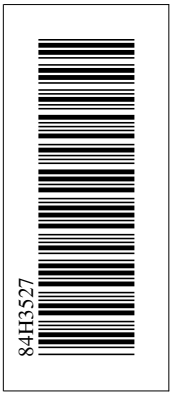
W

width
field choices, setting for 49
fields, setting for 35, 135
section field choices, setting for 60
window header buttons, uses for 11
WTDD
indirection, accomplishing with 128
owner, specifying 6

IBM®

Part Number: 84H3527

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.



SC26-8899-

Spine information:

IBM Intelligent Decision Server Forms Design User's Guide *Version 1*